

TECNOLOGÍAS DE LA INFORMACIÓN EN LINEA

SISTEMAS OPERATIVOS
3 créditos



Profesor Autor: Jose Luis Chauca

Titulaciones	Semestre
<ul style="list-style-type: none">SISTEMAS OPERATIVOS	Segundo

Tutorías: El profesor asignado se publicará en el entorno virtual de aprendizaje (online.utm.edu.ec), y sus horarios de conferencias se indicarán en la sección.

CAFETERÍA VIRTUAL

PERÍODO MAYO 2022/ SEPTIEMBRE 2022

Tabla de contenido

Resultado de aprendizaje de la asignatura	3
Unidad 1: Estructura de los sistemas operativos	3
Tema 1: Visión general del sistema operativo	3
Objetivos y funciones de los sistemas operativos.....	4
Componentes del sistema operativo	14
Niveles de ejecución y secuencia de arranque de los sistemas operativos	16
El GRUB del sistema operativo	39
Interfaz de usuario del sistema operativo.....	41
Tema 2: Estructura de los sistemas operativos.....	43
Estructura simple.....	43
Estructura en niveles.....	43
Clasificación de los sistemas operativos	44
Llamadas al sistema.....	47
Máquinas Virtuales.....	49
Comandos básicos Shell linux.....	53
Bibliografía.....	63



Organización de la lectura para el estudiante por semana del compendio

Semanas	Compendio
Semana 1	Páginas 1 - 14
Semana 2	Páginas 15 - 41
Semana 3	Página 42 – 49
Semana 4	Página 50 – 64



Resultado de aprendizaje de la asignatura

Conocer los conceptos fundamentales, partiendo del enfoque clásico de los sistemas operativos como administrador eficiente de recursos, componentes, estructuras y funciones de los sistemas operativos, con la finalidad que entiendan el funcionamiento y puedan realizar software de sistemas.



SISTEMAS OPERATIVOS



Unidad 1: Estructura de los sistemas operativos

Resultado de aprendizaje de la unidad:

Reconocer los principios generales sobre sistemas operativos de propósito general y de tiempo real, así como los mecanismos básicos de gestión de recursos.



Tema 1: Visión general del sistema operativo

Un sistema operativo es un software que administra el hardware de un computador. Además de proveer un conjunto de aplicaciones básicas que actúan como un intermediario entre el usuario y el hardware de la computadora. Un aspecto interesante de los sistemas operativos es que se encuentran en todas partes, desde un automóvil, y equipos que incluyen dispositivos de "Internet de las Cosas (IoT), hasta teléfonos inteligentes, computadores personales, entornos de informática de la nube, etc., y el cómo cumplen sus tareas en esos entornos tan diversos.

Para entender el papel de los sistemas operativos en un entorno informático actual, es esencial que primero se comprenda la organización y arquitectura del hardware computacional, esta incluye procesador (CPU → **Unidad de Procesamiento Central**), memorias, dispositivos de E/S y almacenamiento. Donde la principal responsabilidad de un sistema operativo es asignar esos recursos a los programas.

Debido a la complejidad de un sistema operativo, este debe ser creado pieza por pieza, donde cada una de estas piezas debe ser diseñada cuidadosamente con entradas, salidas y funciones bien definidas. A lo largo de este primer compendio revisaremos de forma general los componentes principales de un sistema informático actual, así como las funciones que aporta el sistema operativo.



Hardware

• Es toda la parte física del computador, quiere decir aquello que se puede ver y tocar, ej.: procesador, tarjeta madre, tarjetas de sonido, vídeo, memorias, discos duro, etc.

En resumen, El sistema operativo es el único programa que interactúa directamente con el hardware de la computadora, y sus características principales son:

- Abstracción: el programa no se reocupa por los detalles de acceso al hardware, por ejemplo la organización de directorios y archivos (en uno varios dispositivos de almacenamiento)
- Administración de recursos: implementa políticas de asignación de los recursos de forma efectiva de acuerdo a las necesidades de un programa.
- Aislamiento: En un sistema multiusuario y multitarea cada proceso y cada usuario no tendrá que preocuparse por otros que estén usando el mismo sistema

Objetivos y funciones de los sistemas operativos

Para poder definir los objetivos y las funciones de los sistemas operativos contemporáneos, es necesario saber que hacen, entonces:

¿Qué hacen los sistemas operativos?

Primero debemos plantearnos que papel tienen los sistemas operativos en los sistemas informáticos en general, donde un sistema informático puede ser dividido en aproximadamente cuatro componentes: hardware, sistema operativo, programas de aplicación y usuario ver figura 1.



Figura 1: Componentes del sistema operativo.

- El **hardware**, está compuesto por la unidad de procesamiento central (CPU), las memorias, y los dispositivos de entrada y salida (E/S), los cuales proveen los recursos informáticos esenciales para el sistema.

- Los **programas de aplicación** son los recursos utilizados para resolver los problemas informáticos de los **usuarios**, estos pueden ser los procesadores de palabras, las hojas de cálculo, navegadores web, juegos, etc., los actuales además definen cual es el uso principal de sistema informático.
- El **Sistema operativo**, controla el **hardware** y coordina su uso entre los varios **programas de aplicación** para varios **usuarios**.

También se puede considerar que un sistema informático consta de hardware, software y datos. Entonces el sistema operativo provee los medios para el uso apropiado de estos recursos en el funcionamiento del sistema informático. Por lo cual un sistema operativo cumple funciones similares a la de un administrador, porque al igual que este, no realiza funciones por si mismo, sino que provee un entorno en el cual otros programas pueden realizar su trabajo.

Para entender mejor el papel que tiene un sistema operativo, a continuación, se lo analizará desde las perspectivas del usuario y del sistema:

Perspectiva del usuario:

Desde la perspectiva del usuario, la vista del computador varía según la interfaz que se utilice y el equipo que se utilice, sea este un computador portátil o de escritorio, el cual está diseñado para que sólo un usuario esté en control total del dispositivo y sus recursos. De ahí que su objetivo es maximizar el uso que le esté dando el usuario, sea este para trabajar o jugar. Para el caso, el sistema operativo está diseñado principalmente para facilitar su uso, prestando cierta atención al rendimiento y la seguridad y ninguna a la utilización de recursos de cómo se comparten los diversos recursos de hardware y software.

Ahora en la actualidad el incremento en el uso de dispositivos móviles como los teléfonos inteligentes, las tabletas, las consolas de videojuegos, etc., por parte de los usuarios, los cuales están reemplazando a los sistemas informáticos de escritorio y portátiles. Dispositivos que están continuamente conectados a redes celulares o de tecnologías inalámbricas. Cuya interfaz de usuario es típicamente una pantalla táctil, donde el usuario interactúa con el sistema por presionar y deslizar los dedos a través de la pantalla y rara vez utilizan un teclado o un ratón físico, adicionalmente estos dispositivos permiten al usuario interactuar con ellos a través del reconocimiento de voz como Siri de Apple, Cortana de Microsoft, Alexa de Amazon, y el Asistente de Google. De igual forma muchos sistemas pueden o no tener pequeñas interfaces de usuarios como por ejemplo los computadores embebidos de los dispositivos del hogar o automóviles, los cuales muchas veces son solo un teclado con pocos números (ejemplo el control de Chevystar), los cuales al ser activados pueden encender un indicador luminoso o emitir un sonido, para indicar su estado, donde sus aplicaciones y sistemas operativos, están diseñados para ejecutarse sin que el usuario intervenga.

Perspectiva del sistema:

Desde la perspectiva del sistema, el sistema operativo es el programa (porque es un programa) que está más íntimamente involucrado con el hardware. En este contexto el sistema operativo actúa como un gestor de recursos, ya que un sistema informático consta de muchos recursos los cuales pueden ser requeridos para resolver un problema: tiempos del CPU, espacio de memoria, espacio de almacenamiento, dispositivos de E/S, y demás, por lo cual el sistema operativo actúa

como un administrador, gestionando esos recursos, al hacer frente a numerosas y posibles solicitudes conflictivas de recursos, el sistema operativo debe decidir que hacer y como asignarles programas y usuarios específicos para que el sistema informático pueda operar de manera eficiente y correcta.

Una vista ligeramente diferente de un sistema operativo enfatiza la necesidad de controlar los diversos dispositivos de E/S y programas de usuario. Un sistema operativo es un programa de control, el cual gestiona la ejecución de los programas del usuario para evitar errores y un uso inadecuado de la computadora, además se ocupa especialmente del funcionamiento y control de dispositivos de E/S.

Definiendo al sistema operativo:

Hasta ahora se puede apreciar que el termino *sistema operativo* abarca muchas funciones y roles, lo cual es en parte, y esto, debido al gran número de diseños y usos de las computadoras, ya que estas se encuentran presentes en tostadoras, cafeteras, automóviles, barcos, aviones, satélites, naves espaciales, hogares y empresas. De hecho, son la base de las consolas de videojuegos, sintonizadores de canales, sistemas de control industrial, etc.

Para explicar esta diversidad, de debe regresar en la historia de las computadoras, pues a pesar de tener una muy corta historia, estas se han desarrollado muy rápido, de hecho, la informática comenzó como un experimento para determinar lo que se podía hacer y rápidamente se trasladó a sistemas de propósito específicos en área de la milicia, descifrando códigos, trazando trayectorias de misiles, y el uso gubernamental para realizar censos, luego se volvió de uso general, en grandes computadores llamadas *mainframes* que eran multifuncionales, y es ahí donde nace el sistema operativo. Luego en los años 60, Gordon Moore propuso su ley (Moore's Law) la cual predecía que el número de transistores en un circuito integrado se duplicaría cada 18 meses, la cual se ha mantenido cierto, las computadoras ganaron en funcionabilidad y en reducción de su tamaño, llevando estas a un gran número de usos y de sistemas operativos adaptados acá uno de ellos.

Por esto, en general no se puede llegar a definir completamente un sistema operativo, y se debe a que el sistema operativo existe para ofrecer una razonable forma de resolver el problema de crear un sistema informático utilizable.

De ahí que el principal objetivo de un sistema informático es ejecutar programas y hacer que la solución a los problemas sea más fácil. El hardware de la computadora está construido hacia este objetivo. Y como el hardware por sí solo no es fácil de usar, se han desarrollado aplicaciones o programas, los cuales requieren ciertas operaciones comunes como las que controlan los dispositivos de E/S, estas funciones comunes de control y asignación de recursos se han unido en una sola pieza de software: *el sistema operativo*. Se debe recalcar de que no hay una definición universalmente aceptada de lo que es u no es parte de un sistema operativo. Una definición más común, es que el sistema operativo es el **único** programa que se ejecuta en todo momento en la computadora, al cual generalmente se lo llama núcleo (*kernel*).

De ahí que el principal objetivo de un sistema informático es ejecutar programas y hacer que la solución a los problemas sea más fácil.

Junto con el núcleo, existen otros dos tipos de programas: los programas del sistema, que están asociados con el sistema operativo, pero no son necesariamente parte del núcleo, y los programas de aplicación, que incluyen todos los programas que no están asociados con el funcionamiento del sistema. El problema de que funcionalidades forman o no parte de un sistema operativo se volvió cada vez más importante a medida que las computadoras personales se generalizaban y los sistemas operativos se volvían cada vez más sofisticados. Un caso particular de este tema, fue en 1998, cuando el Departamento de Justicia de los Estados Unidos presentó una demanda contra Microsoft, en esencia alegando que Microsoft incluía demasiada funcionalidad en su sistema operativo y, por lo tanto, impedía que los proveedores de aplicaciones compitieran, donde el principal tema de discusión fue porque Microsoft Windows incluía el navegador web Internet Explorer como parte integral de su sistema operativo, como resultado, Microsoft fue declarado culpable de utilizar el monopolio de su sistema operativo para limitar la competencia. Actualmente en 2017, la firma de seguridad Kaspersky Lab, demanda nuevamente a Microsoft por prácticas monopolísticas con su antivirus Windows Defender. Por tales hechos en algunos países de la comunidad europea se vende Microsoft Windows 10 N y en Corea, Windows 10 KN, los cuales quitan todo aquel software que puede ser reemplazado por otras alternativas.

Middleware: *software que actúa como puente entre un sistema operativo o base de datos y las aplicaciones, especialmente en una red.*

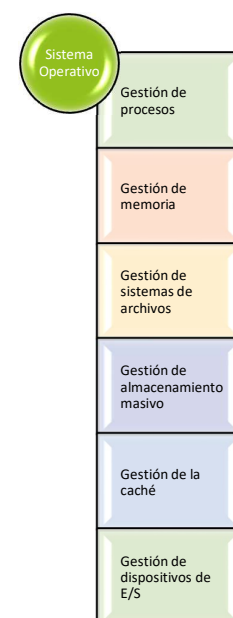
Framework: *es una abstracción en la que el software que proporciona una funcionalidad genérica se puede cambiar de forma selectiva mediante código adicional escrito por el usuario, proporcionando así software específico de la aplicación*

Hoy, sin embargo, si nos fijamos en los sistemas operativos para dispositivos móviles, vemos que una vez más aumenta el número de funciones que constituyen el sistema operativo. Los sistemas operativos móviles a menudo incluyen no solo un núcleo central, sino también *middleware*, un conjunto de software *frameworks* que brindan servicios adicionales a los desarrolladores de aplicaciones. Por ejemplo, cada uno de los dos sistemas operativos móviles más destacados, iOS de Apple y Android de Google, presenta un núcleo central junto con middleware que admite bases de datos, multimedia y gráficos (por nombrar solo algunos).

En resumen, para nuestros propósitos, el sistema operativo incluye el núcleo siempre en ejecución, *middleware frameworks* que facilitan el desarrollo de aplicaciones y proporcionan funciones y programas del sistema que ayudan a administrar el sistema mientras está en ejecución. La mayor parte de este texto se ocupa del núcleo de los sistemas operativos de propósito general, pero se analizan otros componentes según sea necesario para explicar completamente el diseño y la operación del sistema operativo.

Funciones de los sistemas operativos

Como se ha tratado hasta ahora la única función de un sistema operativo es la *gestión de recursos*, como: la CPU del sistema, el espacio de memoria, el espacio de almacenamiento de archivos y los dispositivos de E/S se encuentran entre los recursos que el



sistema operativo debe administrar. De ahí que un sistema operativo gestiona: procesos, memoria, sistemas de archivos o ficheros, almacenamiento masivo, caché y dispositivos de E/S.

Gestión de procesos:

Aclaremos algo, un programa no puede hacer nada, a menos que sus instrucciones sean ejecutadas por una CPU.

Entonces un programa en ejecución es un proceso, sea este un juego, un compilador, un navegador web, etc., mientras se esté ejecutando es un proceso, así como una aplicación de un dispositivo móvil, cuando se está ejecutando, esta es un proceso. Por ahora, se puede considerar un proceso como una instancia de un programa en ejecución, pero más adelante se revisará nuevamente este concepto y se verá que este es más general. Como se describe en la siguiente unidad, es posible proporcionar llamadas al sistema que permitan a los procesos crear subprocesos para ejecutarse simultáneamente.

Un proceso necesita ciertos recursos, incluido el tiempo de CPU, la memoria, los archivos y los dispositivos de E/S, para realizar su tarea. Estos recursos generalmente se asignan al proceso mientras se ejecuta. Además de los diversos recursos físicos y lógicos que obtiene un proceso cuando se crea, se pueden pasar varios datos de inicialización (entrada). Por ejemplo, considere un proceso que ejecuta un navegador web cuya función es mostrar el contenido de una página web en una pantalla. El proceso recibirá la URL como entrada y ejecutará las instrucciones apropiadas y las llamadas al sistema para obtener y mostrar la información deseada en la pantalla. Cuando finaliza el proceso, el sistema operativo recuperará los recursos reutilizables.

Enfatizamos que un programa en sí mismo no es un proceso. Un programa es una entidad pasiva, como el contenido de un archivo almacenado en el disco, mientras que un proceso es una entidad activa. Un proceso de un solo subproceso tiene un contador de programa que especifica la siguiente instrucción a ejecutar. (Los temas se tratan en la siguiente unidad) La ejecución de dicho proceso debe ser secuencial. La CPU ejecuta una instrucción del proceso tras otra, hasta que el proceso se completa. Además, en cualquier momento, se ejecuta una instrucción como máximo en favor del proceso.

El sistema operativo es responsable de las siguientes actividades en relación con la gestión de procesos:

- Crear y eliminar procesos de usuario y del sistema
- Programar procesos e hilos en las CPU
- Suspende y reanuda procesos
- Proporcionar mecanismos para la sincronización de procesos.
- Proporcionar mecanismos para la comunicación de procesos.

Gestión de memoria:

Es importante conocer que la memoria principal es fundamental para el funcionamiento de un sistema informático moderno. La memoria principal es un gran arreglo de bytes, que varían en tamaño desde cientos de miles hasta miles de millones. Cada byte tiene su propia dirección. La memoria principal es un depósito de datos de acceso rápido compartidos por la CPU y los dispositivos de E/S. La CPU lee instrucciones de la memoria principal durante el ciclo de

búsqueda de instrucciones y lee y escribe datos de la memoria principal durante el ciclo de búsqueda de datos (en una arquitectura de von Neumann). La memoria principal es generalmente el único dispositivo de almacenamiento grande al que la CPU puede direccionar y acceder directamente. Por ejemplo, para que la CPU procese datos del disco, esos datos deben transferirse primero a la memoria principal mediante llamadas de E/S generadas por la CPU. De la misma forma, las instrucciones deben estar en la memoria para que la CPU las ejecute.

Para que se ejecute un programa, debe asignarse a direcciones absolutas y cargarse en la memoria. A medida que el programa se ejecuta, accede a las instrucciones y datos del programa desde la memoria generando estas direcciones absolutas. Finalmente, el programa termina, su espacio de memoria se declara disponible y el siguiente programa se puede cargar y ejecutar.

Para mejorar tanto la utilización de la CPU como la velocidad de respuesta de la computadora a sus usuarios, las computadoras de uso general deben mantener varios programas en la memoria, lo que crea la necesidad de administrar la memoria. Se utilizan muchos esquemas de administración de memoria diferentes. Estos esquemas reflejan varios enfoques y la efectividad de cualquier algoritmo dado depende de la situación. Al seleccionar un esquema de administración de memoria para un sistema específico, debemos tener en cuenta muchos factores, especialmente el diseño de hardware del sistema. Cada algoritmo requiere su propio soporte de hardware.

El sistema operativo es responsable de las siguientes actividades relacionadas con la gestión de la memoria:

- Realizar un seguimiento de qué partes de la memoria se están utilizando actualmente y qué proceso las está utilizando
- Asignar y desasignar espacio de memoria según sea necesario
- Decidir qué procesos (o partes de procesos) y datos se moverán hacia y desde la memoria

Gestión del sistema de archivos:

Para que el sistema informático sea conveniente para los usuarios, el sistema operativo proporciona una vista lógica y uniforme del almacenamiento de información. El sistema operativo se abstrae de las propiedades físicas de sus dispositivos de almacenamiento para definir una unidad lógica de almacenamiento, el sistema de archivos. El sistema operativo asigna archivos a medios físicos y accede a estos archivos a través de los dispositivos de almacenamiento.

La gestión de archivos es uno de los componentes más visibles de un sistema operativo. Las computadoras pueden almacenar información en varios tipos diferentes de medios físicos. El almacenamiento secundario es el más común, pero también es posible el almacenamiento terciario. Cada uno de estos medios tiene sus propias características y organización física. La mayoría están controlados por un dispositivo, como una unidad de disco, que también tiene sus propias características únicas. Estas propiedades incluyen la velocidad de acceso, la capacidad, la tasa de transferencia de datos y el método de acceso (secuencial o aleatorio).

Un archivo es una colección de información relacionada definida por su creador. Por lo general, los archivos representan programas (fuentes y formas de objeto) y datos. Los archivos de datos

pueden ser numéricos, alfabéticos, alfanuméricos o binarios. Los archivos pueden ser de formato libre (por ejemplo, archivos de texto) o pueden tener un formato rígido (por ejemplo, campos fijos como un archivo de música mp3). Claramente, el concepto de archivo es extremadamente general.

El sistema operativo implementa el concepto abstracto de un archivo al administrar los medios de almacenamiento masivo y los dispositivos que los controlan. Además, los archivos normalmente se organizan en directorios para facilitar su uso. Finalmente, cuando varios usuarios tienen acceso a archivos, puede ser conveniente controlar qué usuario puede acceder a un archivo y cómo ese usuario puede acceder a él (por ejemplo, leer, escribir, agregar).

El sistema operativo es responsable de las siguientes actividades relacionadas con la gestión de archivos:

- Crear y eliminar archivos
- Crear y eliminar directorios para organizar archivos.
- Soporta primitivas para manipular archivos y directorios.
- Asignación de archivos al almacenamiento masivo
- Hacer copias de seguridad de archivos en medios de almacenamiento estables (no volátiles)

Gestión de almacenamiento masivo:

Como ya hemos visto, el sistema informático debe proporcionar almacenamiento secundario para respaldar la memoria principal. La mayoría de los sistemas informáticos modernos utilizan unidad de discos duros (*HDD* → **Hard Disk Drive**) y dispositivos de memoria no volátil (*NVM* → **Non-Volatile Memory**) como el principal medio de almacenamiento en línea para programas y datos.

La mayoría de los programas, incluidos compiladores, navegadores web, procesadores de texto y juegos, se almacenan en estos dispositivos hasta que se cargan en la memoria. Luego, los programas utilizan los dispositivos como fuente y destino de su procesamiento.

Por tanto, la gestión adecuada del almacenamiento secundario es de vital importancia para un sistema informático. El sistema operativo es responsable de las siguientes actividades en relación con la gestión del almacenamiento secundario:

- Montaje y desmontaje
- Gestión del espacio libre
- Asignación de almacenamiento
- Planificación de disco
- Fraccionamiento o particionamiento
- Protección

Debido a que el almacenamiento secundario se usa con frecuencia y de manera extensiva, debe usarse de manera eficiente. Toda la velocidad de funcionamiento de una computadora puede depender de las velocidades del subsistema de almacenamiento secundario y de los algoritmos que manipulan ese subsistema.

Al mismo tiempo, hay muchos usos para el almacenamiento que son más lentos y de menor costo (y, a veces, de mayor capacidad) que el almacenamiento secundario. Las copias de seguridad de los datos del disco, el almacenamiento de datos poco utilizados y el almacenamiento de archivos a largo plazo son algunos ejemplos. Las unidades de cinta magnética y sus cintas y las unidades y platos de CD, DVD y Blu-ray son dispositivos de almacenamiento terciario típicos.

El almacenamiento terciario no es crucial para el rendimiento del sistema, pero aún debe administrarse. Algunos sistemas operativos se encargan de esta tarea, mientras que otros dejan la gestión del almacenamiento terciario a los programas de aplicación. Algunas de las funciones que pueden proporcionar los sistemas operativos incluyen el montaje y desmontaje de medios en dispositivos, la asignación y liberación de dispositivos para uso exclusivo de los procesos y la migración de datos del almacenamiento secundario al terciario.

Gestión de caché:

El almacenamiento en caché es un principio importante de los sistemas informáticos. Así es como funciona. La información normalmente se guarda en algún sistema de almacenamiento (como la memoria principal). A medida que se utiliza, se copia en un sistema de almacenamiento más rápido, *el caché*, de forma temporal. Cuando necesitamos una información en particular, primero verificamos si está en la caché. Si es así, usamos la información directamente del caché. Si no es así, usamos la información de la fuente, poniendo una copia en el caché bajo el supuesto de que la necesitaremos nuevamente pronto.

Además, los registros programables internos proporcionan una caché de alta velocidad para la memoria principal. El programador (o compilador) implementa la asignación de registros y los algoritmos de reemplazo de registros para decidir qué información conservar en los registros y cuál en la memoria principal.

Otros cachés se implementan totalmente en hardware. Por ejemplo, la mayoría de los sistemas tienen un caché de instrucciones para almacenar las instrucciones que se espera que se ejecuten a continuación. Sin esta caché, la CPU tendría que esperar varios ciclos mientras se recupera una instrucción de la memoria principal. Por razones similares, la mayoría de los sistemas tienen una o más cachés de datos de alta velocidad en la jerarquía de memoria. No nos preocupan estos cachés de hardware, ya que están fuera del control del sistema operativo.

Debido a que las cachés tienen un tamaño limitado, la administración de la caché es un problema de diseño importante. La selección cuidadosa del tamaño de la caché y de una política de reemplazo puede resultar en un rendimiento mucho mayor, como puede ver al examinar la Tabla 1.

El movimiento de información entre niveles de una jerarquía de almacenamiento puede ser explícito o implícito, según el diseño del hardware y el software del sistema operativo de control. Por ejemplo, la transferencia de datos de la caché a la CPU y los registros suele ser una función de hardware, sin intervención del sistema operativo. Por el contrario, la transferencia de datos del disco a la memoria suele estar controlada por el sistema operativo.

Tabla 1: Características de varios tipos de almacenamientos

Nivel	1	2	3	4	5
Nombre	registros	caché	memoria principal	disco de estado solido (SSD)	disco magnético
Tamaño típico	< 1KB	>16 MB	< 65 GB	< 1TB	< 10 TB
Implementación tecnológica	memoria personalizada con múltiples puertos CMOS	CMOS SRAM en chip o fuera de chip	CMOS SRAM	memoria tipo flash	disco magnético
Tiempo de acceso (ns)	0.25 – 0.50	0.50 – 25	80 – 250	25,000 – 50,000	5'000,000
Ancho de banda (MB/sec)	20,000 – 100,000	5,000 – 10,000	1,000 – 5,000	500	20 – 150
Gestionado por	compilador	hardware	sistema operativo	sistema operativo	sistema operativo
Respaldo por	cache	memoria principal	disco	disco	disco o cinta

En una estructura de almacenamiento jerárquica, los mismos datos pueden aparecer en diferentes niveles del sistema de almacenamiento. Por ejemplo, suponga que un entero **A** que se va a incrementar en 1 está ubicado en el archivo **B** y el archivo **B** reside en el disco duro. La operación de incremento procede emitiendo primero una operación de E/S para copiar el bloque de disco en el que **A** reside a la memoria principal. A esta operación le sigue la copia de **A** en la caché y en un registro interno. Así, la copia de **A** aparece en varios lugares: en el disco duro, en la memoria principal, en la caché y en un registro interno (ver Figura 2). Una vez que se produce el incremento en el registro interno, el valor de **A** difiere en los distintos sistemas de almacenamiento. El valor de **A** se convierte en el mismo sólo después de que el nuevo valor de **A** se escribe desde el registro interno de nuevo al disco duro.

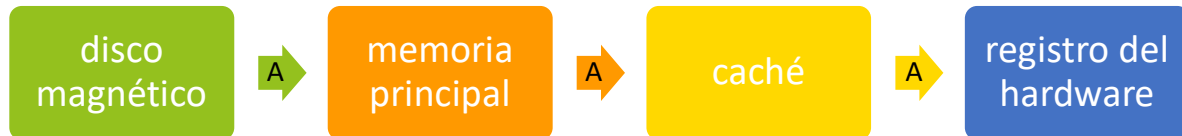


Figura 2: Migración del entero **A** desde el disco al registro

En un entorno informático donde solo se ejecuta un proceso a la vez, esta disposición no plantea dificultades, ya que un acceso al entero **A** siempre será a la copia en el nivel más alto de la jerarquía. Sin embargo, en un entorno multitarea, donde la CPU se alterna entre varios procesos, se debe tener sumo cuidado para garantizar que, si varios procesos desean acceder a **A**, cada uno de estos procesos obtendrá el valor actualizado más recientemente de **A**.

La situación se vuelve más complicada en un entorno multiprocesador donde, además de mantener registros internos, cada una de las CPU también contiene una caché local (será tratado en el siguiente compendio). En tal entorno, una copia de **A** puede existir simultáneamente en varias cachés. Dado que las distintas CPU pueden ejecutarse todas en paralelo, debemos asegurarnos de que una actualización del valor de **A** en una caché se refleje inmediatamente en todas las demás cachés donde **A** reside. Esta situación se denomina coherencia de caché y suele ser un problema de hardware (que se maneja por debajo del nivel del sistema operativo).

En un entorno distribuido, la situación se vuelve aún más compleja. En este entorno, se pueden guardar varias copias (o réplicas) del mismo archivo en diferentes computadoras. Dado que se puede acceder y actualizar las diversas réplicas al mismo tiempo, algunos sistemas distribuidos garantizan que, cuando una réplica se actualiza en un lugar, todas las demás réplicas se actualizan lo antes posible, para lograrlo hay varias formas de lograr esta garantía (este tema no será tratado en esta asignatura).

Gestión del sistema de E/S:

Uno de los propósitos de un sistema operativo es ocultar al usuario las peculiaridades de dispositivos de hardware específicos. Por ejemplo, en UNIX, el subsistema de E/S oculta las peculiaridades de los dispositivos de E/S a la mayor parte del sistema operativo. El subsistema de E/S consta de varios componentes:

- Un componente de administración de memoria que incluye almacenamiento en búfer, almacenamiento en caché y cola
- Una interfaz general de controlador de dispositivo
- Controladores para dispositivos de hardware específicos

Solo el controlador del dispositivo conoce las peculiaridades del dispositivo específico al que está asignado.

Para esto, el sistema operativo se apoya en la manipulación de las interrupciones y los controladores de dispositivos los cuales son usados para la construcción de subsistemas de E/S eficientes.

Seguridad y protección:

Si un sistema informático tiene múltiples usuarios y permite la ejecución concurrente de múltiples procesos, entonces el acceso a los datos debe estar regulado. Para ese propósito, los mecanismos aseguran que los archivos, segmentos de memoria, CPU y otros recursos puedan ser operados por solo aquellos procesos que hayan obtenido la autorización adecuada del sistema operativo. Por ejemplo, el hardware de direccionamiento de memoria asegura que un proceso se pueda ejecutar solo dentro de su propio espacio de direcciones. El temporizador asegura que ningún proceso pueda obtener el control de la CPU sin renunciar finalmente al control. Los usuarios no pueden acceder a los registros de control de dispositivos, por lo que la integridad de los distintos dispositivos periféricos está protegida. La protección, entonces, es cualquier mecanismo para controlar el acceso a los procesos.

La protección, entonces, es cualquier mecanismo para controlar el acceso de procesos o usuarios a los recursos definidos por un sistema informático. Este mecanismo debe proporcionar medios para especificar los controles que se impondrán y hacer cumplir los controles.

La protección puede mejorar la confiabilidad al detectar errores latentes en las interfaces entre los subsistemas de componentes. La detección temprana de errores de interfaz a menudo puede evitar la contaminación de un subsistema en buen estado por otro subsistema que no funciona correctamente. Además, un recurso desprotegido no puede defenderse del uso (o mal uso) por parte de un usuario no autorizado o incompetente. Un sistema orientado a la protección proporciona un medio para distinguir entre el uso autorizado y no autorizado.

Un sistema puede tener la protección adecuada, pero aun así ser propenso a fallar y permitir un acceso inadecuado. Considere un usuario cuya información de autenticación (su medio de identificarse en el sistema) es robada. Sus datos se pueden copiar o eliminar, aunque la protección de archivos y memoria esté funcionando. El trabajo de la seguridad es defender un sistema de ataques externos e internos. Dichos ataques se extienden por una amplia gama e incluyen virus y gusanos, ataques de denegación de servicio (que usan todos los recursos del sistema y, por lo tanto, mantienen a los usuarios legítimos fuera del sistema), robo de identidad y robo de servicio (uso no autorizado de un sistema). La prevención de algunos de estos ataques se considera una función del sistema operativo en algunos sistemas, mientras que otros sistemas lo dejan en manos de políticas o software adicional. Debido al alarmante aumento de los incidentes de seguridad, las características de seguridad del sistema operativo son un área de investigación e implementación de rápido crecimiento.

La protección y la seguridad requieren que el sistema pueda distinguir entre todos sus usuarios. La mayoría de los sistemas operativos mantienen una lista de nombres de usuario y un identificador de usuario asociado (ID de usuario). En lenguaje de Windows, este es un ID de seguridad (SID). Estos ID numéricos son únicos, uno por usuario. Cuando un usuario inicia sesión en el sistema, la etapa de autenticación determina el ID de usuario apropiado para el usuario. Ese ID de usuario está asociado con todos los procesos e hilos del usuario. Cuando un usuario necesita leer una identificación, se vuelve a traducir al nombre de usuario a través de la lista de nombres de usuario.

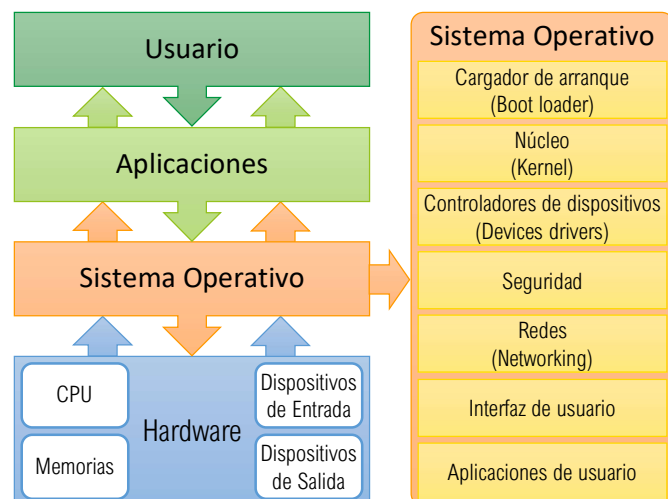
En algunas circunstancias, deseamos distinguir entre conjuntos de usuarios en lugar de usuarios individuales. Por ejemplo, al propietario de un archivo en un sistema UNIX se le puede permitir ejecutar todas las operaciones en ese archivo, mientras que a un grupo seleccionado de usuarios solo se le puede permitir leer el archivo. Para lograr esto, necesitamos definir un nombre de grupo y el conjunto de usuarios que pertenecen a ese grupo. La funcionalidad de grupo se puede implementar como una lista de todo el sistema de nombres de grupo e identificador de grupo. Un usuario puede estar en uno o más grupos, según las decisiones de diseño del sistema operativo. Los ID de grupo del usuario también se incluyen en cada proceso e hilo asociado.

En el curso del uso normal del sistema, el ID de usuario y el ID de grupo de un usuario son suficientes. Sin embargo, un usuario a veces necesita escalar privilegios para obtener permisos adicionales para una actividad. El usuario puede necesitar acceso a un dispositivo que está restringido, por ejemplo. Los sistemas operativos proporcionan varios métodos para permitir la escalada de privilegios. En UNIX, por ejemplo, el atributo *setuid* en un programa hace que ese programa se ejecute con el ID de usuario del propietario del archivo, en lugar del ID del usuario actual. El proceso se ejecuta con este UID efectivo hasta que desactiva los privilegios adicionales o termina.

Componentes del sistema operativo

Los componentes de un sistema operativos básicamente son:

Núcleo:



Es un software que constituye una parte fundamental del sistema operativo, y se define como la parte que se ejecuta en modo privilegiado (conocido también como modo núcleo). Es el principal responsable de facilitar a los distintos programas acceso seguro al hardware de la computadora o en forma básica, es el encargado de gestionar recursos, a través de servicios de llamada al sistema. Como hay muchos programas y el acceso al hardware es limitado, también se encarga de decidir qué programa podrá usar un dispositivo de hardware y durante cuánto tiempo, lo que se conoce como multiprogramación. Acceder al hardware directamente puede ser realmente complejo, por lo que los núcleos suelen implementar una serie de abstracciones del hardware. Esto permite esconder la complejidad, y proporcionar una interfaz limpia y uniforme al hardware subyacente, lo que facilita su uso al programador.

Figura 3: Componentes de un sistema operativo

En algunos sistemas operativos, no existe un núcleo como tal (algo común en sistemas embebidos), debido a que en ciertas arquitecturas no hay distintos modos de ejecución.

Cargador de arranque:

Un cargador de arranque carga un sistema operativo en la memoria, realiza la inicialización y comienza la ejecución del sistema.

Este permite seleccionar la partición raíz la cual contiene el núcleo del sistema operativo y a veces otros archivos del sistema.

Controladores de dispositivos:

Normalmente, los sistemas operativos tienen un software controlador de dispositivo (*device driver*) para cada controlador electrónico de dispositivo (*device controller*). Este software controlador de dispositivo comprende el controlador electrónico de dispositivo y proporciona al resto del sistema operativo una interfaz uniforme para el dispositivo. La CPU y los controladores electrónicos de dispositivos pueden ejecutarse en paralelo, compitiendo por ciclos de memoria. Para garantizar el acceso ordenado a la memoria compartida, un controlador electrónico de memoria sincroniza el acceso a la memoria.

Los softwares controladores de dispositivos permiten habilitar las operaciones de E/S de un sistema, porque un programa en ejecución puede requerir E/S, lo que puede implicar un archivo o un dispositivo de E/S. Para dispositivos específicos, se pueden desear funciones especiales (como leer desde una interfaz de red o escribir en un sistema de archivos). Por razones de eficiencia y protección, los usuarios generalmente no pueden controlar los dispositivos de E/S directamente. Por lo tanto, el sistema operativo a través de estos softwares controladores de dispositivos puede proporcionar un medio para realizar las actividades de E/S.

Seguridad:

La importancia de la protección y seguridad hace que los propietarios de la información almacenada en un sistema informático multiusuario o en red pueden querer controlar el uso de esa información. Cuando varios procesos separados se ejecutan al mismo tiempo, no debería ser posible que un proceso interfiera con los demás o con el sistema operativo en sí. La protección implica garantizar que se controle todo el acceso a los recursos del sistema. La seguridad del sistema frente a personas externas también es importante. Dicha seguridad comienza requiriendo que cada usuario se autentique en el sistema, generalmente mediante una

contraseña, para obtener acceso a los recursos del sistema. Se extiende a la defensa de los dispositivos de E/S externos, incluidos los adaptadores de red, de los intentos de acceso no válidos y al registro de todas esas conexiones para la detección de robos. Para proteger y asegurar un sistema, se deben tomar precauciones a través de este. Una cadena es tan fuerte como su eslabón más débil.

Redes:

Una red, en los términos más simples, es una ruta de comunicación entre dos o más sistemas. Los sistemas distribuidos dependen de las redes para su funcionalidad. Las redes varían según los protocolos utilizados, las distancias entre los nodos y los medios de transporte. TCP/IP es el protocolo de red más común y proporciona la arquitectura fundamental de Internet. La mayoría de los sistemas operativos admiten TCP/IP, incluidos todos los de uso general. Algunos sistemas admiten protocolos propietarios para adaptarse a sus necesidades. Para un sistema operativo, solo es necesario que un protocolo de red tenga un dispositivo de interfaz (un adaptador de red, por ejemplo) con un controlador de dispositivo para administrarlo, así como un software para manejar datos.

Interfaz de usuario:

Casi todos los sistemas operativos tienen una interfaz de usuario (UI).

Esta interfaz puede adoptar varias formas. Por lo general, se utiliza una interfaz gráfica de usuario (*GUI* → *Graphic User Interface*). Aquí, la interfaz es un sistema de ventana con un ratón que sirve como un dispositivo señalador para dirigir E/S, elegir entre menús y hacer selecciones y un teclado para ingresar texto. Los sistemas móviles, como teléfonos y tabletas, proporcionan una interfaz de pantalla táctil que permite a los usuarios deslizar los dedos por la pantalla o presionar botones en la pantalla para seleccionar opciones. Otra opción es una interfaz de línea de comandos (*CLI* → *Command Line Interface*), que usa comandos de texto y un método para ingresarlos (por ejemplo, un teclado para ingresar comandos en un formato específico con opciones específicas). Algunos sistemas proporcionan dos o las tres variaciones.

Aplicaciones de usuario:

La ejecución de programa o aplicaciones de usuarios es una parte fundamental de los sistemas operativos, porque el sistema debe poder cargar un programa en la memoria y ejecutar ese programa. El programa debe poder finalizar su ejecución, ya sea normal o anormalmente (indicando error).

Niveles de ejecución y secuencia de arranque de los sistemas operativos

El proceso de arranque de un sistema operativo Linux se inicializa de la siguiente manera:

Cuando usted enciende su servidor o su computadora personal, esta hace que el BIOS de su equipo inicie las operaciones relacionadas con el arranque. El **BIOS** (Basic Input Output System) es un pequeño programa escrito en lenguaje ensamblador cuya función es cargar el sistema operativo en la memoria RAM (Random Access Memory), una vez que el BIOS carga el sistema operativo en RAM este inicia un proceso llamado **POST** (Power On Self Test) el cual es un proceso de diagnóstico y verificación de los componentes de entrada y salida de un servidor o computadora y se encarga de configurar y diagnosticar el estado del hardware, una vez verificado el hardware se inicia la fase de arranque del sistema (**bootstrapping**) el cual cede el

control al GRUB (Grand Unified Bootloader), el **GRUB** es un gestor de arranque que hace uso de un menú gráfico que permite elegir el Sistema Operativo que se desea arrancar; Así mismo, el GRUB realiza las siguientes tareas:

- ✓ Cargar el kernel en memoria.
- ✓ Cargar el sistema de ficheros virtual **initrd** el cual es usado típicamente para hacer los arreglos necesarios antes de que el sistema de ficheros raíz pueda ser montado.
- ✓ Pasarle los argumentos **runlevel** e **init** al kernel
- ✓ Comenzar la ejecución del **kernel**



*Al terminar de ejecutar todas las tareas anteriores el GRUB le cede el control total del arranque al kernel y este a su vez se encarga de realizar la llamada a la función **starup** la cual tiene como función detectar el tipo de CPU con el que el equipo cuenta así como de lo principal del sistema operativo, como el manejo de memoria, planificador de tareas, entradas y salidas, comunicación interprocesos, y demás sistemas de control, a partir de este momento se ejecuta el proceso **INIT**.*

El Proceso INIT

INIT es el primer proceso en ejecutarse después de la carga del kernel de Linux e implementa dos modelos bajo los cuales puede trabajar, estos son

- ✓ SystemV
- ✓ BSD

Estos modelos son arrancados por un programa (script) de arranque que establece como deben inicializarse los diferentes servicios, programas o registros que sean necesarios para que el sistema funcione como el administrador lo requiere.

Explicaremos brevemente como es que trabajan estos modelos

SystemV

Es un modelo usado para controlar el inicio y apagado del sistema y fue originalmente desarrollado por la compañía estadounidense de telecomunicaciones AT&T.

SystemV fue una de las versiones del sistema operativo Unix que se encargaba de controlar el arranque de los programas en el instante de inicio del equipo. Este modelo es considerado por muchos como fácil, potente y flexible en comparación con el sistema de inicio BSD.



Existen cuatro versiones reléase de SystemV (**SVR**), las cuales son:

1. **SVR1.** Primera versión de SystemV lanzada en 1984, incluía el editor de textos Vi
2. **SVR2.** Incluye mejoras con respecto al núcleo el cual esta implementado como memoria virtual paginada, el sistema operativo Apple está basado en este modelo.
3. **SVR3.** Incluye mejoras en el sistema de ficheros así como una nueva API de red, el sistema operativo AIX de IBM hace uso de este modelo
4. **SVR4.** Fue la versión más popular de SVR así como la fuente de varias características comunes del sistema operativo Unix, como el script `/etc/init.d`

Los niveles de ejecución en SystemV describen ciertos estados del equipo los cuales se caracterizan por ejecutar ciertos procesos. En general existen 8 niveles de ejecución los cuales van del 0 al 6 y S o s, que son alias del mismo nivel de ejecución, de estos ocho niveles, tres son considerados reservados, estos son:

1. Halt
2. Single user mode
3. Reboot

Aparte de los niveles de ejecución 0,1 y 6 todos los sistemas operativos Linux tratan a los niveles de ejecución un poco diferente. El denominador común de todas las distribuciones Linux es el fichero **`/etc/inittab`**

0 Indica halt o apagado de la máquina.

1 Indica monousuario.

2 Indica modo multiusuario sin soporte de red.

3 Indica modo multiusuario completo con soporte de red.

4 No usado, con esta opción el administrador puede personalizar el inicio para cargar algún servicio.

5 Indica multiusuario completo con inicio gráfico (X11)

6 Indica shutdown y reboot: Se apaga inmediatamente la máquina para reinicio.



Un administrador (root) puede editar el archivo `/etc/inittab` como mejor convenga al usuario, sin embargo también tiene el poder de establecerlo en 0 o en 6.

A continuación un ejemplo de cuantos niveles de ejecución tienen cada una de las distribuciones más importantes de Linux, así como del sistema operativo Solaris y AIX, como muestra la Tabla 3.


Sistema Operativo	Niveles de ejecución por default
	2
	2
 gentoo linux™	3
	5
 redhat	3 o 5
 fedora	3 o 5
	3
	5
	2
 solaris	2

Tabla 3: Niveles de ejecución de los sistemas operativos

En la mayoría de los sistemas operativos Linux los usuarios pueden saber bajo qué nivel de ejecución están trabajando, tecleando en una consola y como root lo siguiente:

[root@localhost ~]# who -r



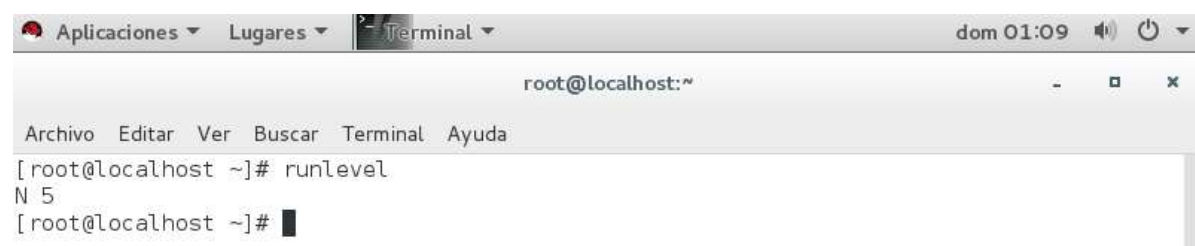
```
Aplicaciones ▾ Lugares ▾ Terminal ▾ dom 01:05 🔊 🔌
root@localhost:~
Archivo Editar Ver Buscar Terminal Ayuda
[root@localhost ~]# who -r
`run-level' 5 2021-04-25 00:33
[root@localhost ~]#
```

Figura 4: Resultado de la ejecución del comando who -r

El **who** y la bandera **-r** nos indican el **runlevel** del equipo y la fecha en la que se inició ese runlevel.

También con el comando **runlevel**

[root@localhost ~]# runlevel



```
Aplicaciones ▾ Lugares ▾ Terminal ▾ dom 01:09 🔊 🔌
root@localhost:~
Archivo Editar Ver Buscar Terminal Ayuda
[root@localhost ~]# runlevel
N 5
[root@localhost ~]#
```

Figura 5: Resultado de la ejecución del comando runlevel

En caso de que se haya cambiado previamente el **runlevel**, **N** indicará el estado anterior del runlevel.

Existen también los ficheros llamados **rcN.d** en donde la letra **N** representa cada uno de los niveles de ejecución en los que trabaja init.d, la función de estos ficheros se explicara más a detalle en el siguiente tema.



*Es importante mencionar que en muchas distribuciones modernas de Linux, se ha cambiado **/etc/inittab** por el uso de **/etc/systemd**, aunque ambas versiones siguen funcionando la **realidad es que inittab dejará de ser el sistema por defecto para darle pie a systemd.***

Systemd

Como se mencionó previamente systemd es el reemplazo para el viejo y tradicional sistema **System V init**, systemd fue diseñado para permitir un manejo de dependencias y tiene la habilidad de manejar más tareas en paralelo al momento en que el sistema inicia. Systemd cuenta con **snapshotting** del sistema y a su vez recuperación del estado del sistema, realiza un seguimiento de los procesos almacenados en lo que se conoce como un **cgroup** en comparación con el método convencional **PID**, systemd se encuentra actualmente por defecto con muchas distribuciones de Linux populares, como Debían, Fedora, Mandriva, Mageia, Ubuntu, Arch Linux, CentOS 7, RHEL 7.0 (Red Hat Enterprise Linux) y Oracle Linux 7.0, **systemd se refiere a los niveles de ejecución como targets**.

Controlando los Runlevel

Para mostrar el runlevel actual de nuestro sistema, lo podemos hacer con la siguiente instrucción:

```
[root@localhost ~]# systemctl get-default
```



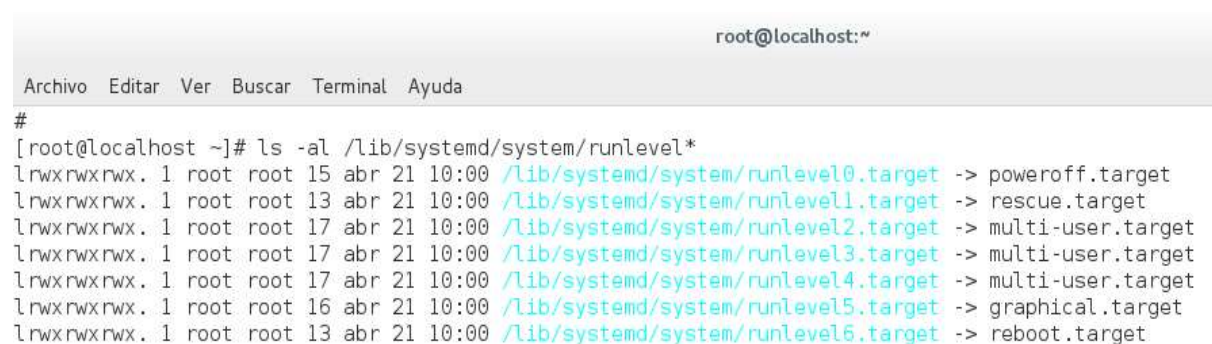
```
root@localhost:~  
Archivo Editar Ver Buscar Terminal Ayuda  
[root@localhost ~]# systemctl get-default  
graphical.target  
[root@localhost ~]#
```

Figura 6: Resultado de la ejecución del comando `systemctl get-default`

Como respuesta tenemos que el sistema se encuentra en **graphical.target**. Básicamente el nivel de ejecución **graphical.target** es el equivalente al tradicional **runlevel 5**, Acceso completo a usuarios con entorno gráfico y red.

Para desplegar los nuevos **runlevels/targets** con el siguiente comando:

```
[root@localhost ~]# ls -al /lib/systemd/system/runlevel*
```



```
root@localhost:~  
Archivo Editar Ver Buscar Terminal Ayuda  
#  
[root@localhost ~]# ls -al /lib/systemd/system/runlevel*  
lrwxrwxrwx. 1 root root 15 abr 21 10:00 /lib/systemd/system/runlevel0.target -> poweroff.target  
lrwxrwxrwx. 1 root root 13 abr 21 10:00 /lib/systemd/system/runlevel1.target -> rescue.target  
lrwxrwxrwx. 1 root root 17 abr 21 10:00 /lib/systemd/system/runlevel2.target -> multi-user.target  
lrwxrwxrwx. 1 root root 17 abr 21 10:00 /lib/systemd/system/runlevel3.target -> multi-user.target  
lrwxrwxrwx. 1 root root 17 abr 21 10:00 /lib/systemd/system/runlevel4.target -> multi-user.target  
lrwxrwxrwx. 1 root root 16 abr 21 10:00 /lib/systemd/system/runlevel5.target -> graphical.target  
lrwxrwxrwx. 1 root root 13 abr 21 10:00 /lib/systemd/system/runlevel6.target -> reboot.target
```

Figura 7: Resultado de la ejecución del comando `ls -al /lib/systemd/system/runlevel*`

De la imagen podemos ver que existen 7 diferentes niveles de ejecución comprendidos entre **system poweroff** y **system reboot**.

```

root@localhost:~
Archivo  Editar  Ver  Buscar  Terminal  Ayuda

#
[root@localhost ~]# ls -al /lib/systemd/system/runlevel*
lrwxrwxrwx. 1 root root 15 abr 21 10:00 /lib/systemd/system/runlevel0.target -> poweroff.target
lrwxrwxrwx. 1 root root 13 abr 21 10:00 /lib/systemd/system/runlevel1.target -> rescue.target
lrwxrwxrwx. 1 root root 17 abr 21 10:00 /lib/systemd/system/runlevel2.target -> multi-user.target
lrwxrwxrwx. 1 root root 17 abr 21 10:00 /lib/systemd/system/runlevel3.target -> multi-user.target
lrwxrwxrwx. 1 root root 17 abr 21 10:00 /lib/systemd/system/runlevel4.target -> multi-user.target
lrwxrwxrwx. 1 root root 16 abr 21 10:00 /lib/systemd/system/runlevel5.target -> graphical.target
lrwxrwxrwx. 1 root root 13 abr 21 10:00 /lib/systemd/system/runlevel6.target -> reboot.target

/lib/systemd/system/runlevel1.target.wants:
total 24
drwxr-xr-x. 2 root root 50 abr 21 10:00 .
drwxr-xr-x. 27 root root 20480 abr 21 10:05 ..
lrwxrwxrwx. 1 root root 39 abr 21 10:00 systemd-update-utmp-runlevel.service -> ../systemd-update-utmp-runlevel.service

/lib/systemd/system/runlevel2.target.wants:
total 24
drwxr-xr-x. 2 root root 50 abr 21 10:00 .
drwxr-xr-x. 27 root root 20480 abr 21 10:05 ..
lrwxrwxrwx. 1 root root 39 abr 21 10:00 systemd-update-utmp-runlevel.service -> ../systemd-update-utmp-runlevel.service

/lib/systemd/system/runlevel3.target.wants:
total 24
drwxr-xr-x. 2 root root 50 abr 21 10:00 .
drwxr-xr-x. 27 root root 20480 abr 21 10:05 ..
lrwxrwxrwx. 1 root root 39 abr 21 10:00 systemd-update-utmp-runlevel.service -> ../systemd-update-utmp-runlevel.service

/lib/systemd/system/runlevel4.target.wants:
total 24
drwxr-xr-x. 2 root root 50 abr 21 10:00 .
drwxr-xr-x. 27 root root 20480 abr 21 10:05 ..
lrwxrwxrwx. 1 root root 39 abr 21 10:00 systemd-update-utmp-runlevel.service -> ../systemd-update-utmp-runlevel.service

/lib/systemd/system/runlevel5.target.wants:
total 24
drwxr-xr-x. 2 root root 50 abr 21 10:00 .
drwxr-xr-x. 27 root root 20480 abr 21 10:05 ..
lrwxrwxrwx. 1 root root 39 abr 21 10:00 systemd-update-utmp-runlevel.service -> ../systemd-update-utmp-runlevel.service
[root@localhost ~]#

```

Figura 8: Resultado de la ejecución del comando `ls -al /lib/systemd/system/runlevel*`

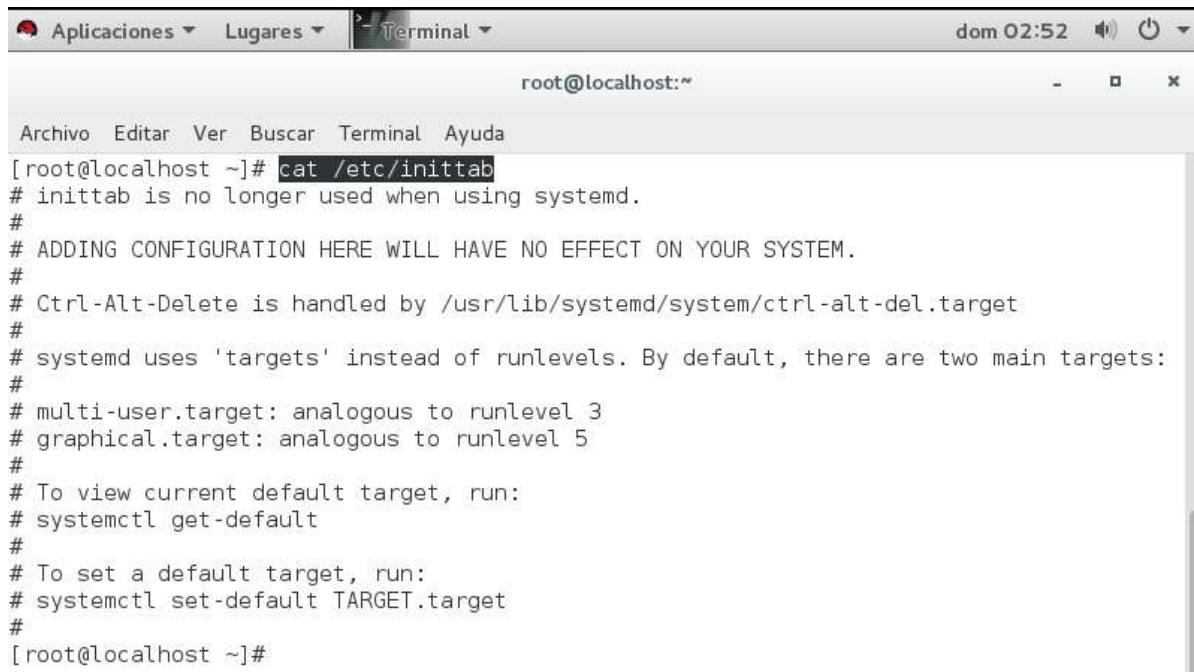
Si hacemos una comparación entre los niveles de ejecución y los targets quedaría algo así como muestra la tabla:

Runlevel	Systemd Description
0	poweroff.target
1	rescue.target
2	multi-user.target
3	multi-user.target
4	multi-user.target
5	graphical.target
6	reboot.target

Tabla 4: Descripción de los niveles de ejecución

El nivel por default se ubicaba dentro del archivo `/etc/inittab` y podía ser impreso en pantalla con la siguiente instrucción: `cat /etc/inittab`

```
[root@localhost ~]# cat /etc/inittab
```

```
root@localhost:~  
Archivo  Editar  Ver  Buscar  Terminal  Ayuda  
[root@localhost ~]# cat /etc/inittab  
# inittab is no longer used when using systemd.  
#  
# ADDING CONFIGURATION HERE WILL HAVE NO EFFECT ON YOUR SYSTEM.  
#  
# Ctrl-Alt-Delete is handled by /usr/lib/systemd/system/ctrl-alt-del.target  
#  
# systemd uses 'targets' instead of runlevels. By default, there are two main targets:  
#  
# multi-user.target: analogous to runlevel 3  
# graphical.target: analogous to runlevel 5  
#  
# To view current default target, run:  
# systemctl get-default  
#  
# To set a default target, run:  
# systemctl set-default TARGET.target  
#  
[root@localhost ~]#
```

Figura 9: Resultado de la ejecución del comando `cat /etc/inittab`

BSD



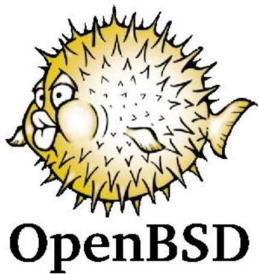
El modelo **BSD init** se ejecuta mediante el script de inicialización situado en la ruta `/etc/rc`

Algunos de los sistemas operativos que se basan en este modelo son los basados en BSD como:

- ✓ FreeBSD
- ✓ NetBSD
- ✓ OpenBSD
- ✓ DragonFlyBSD
- ✓ DesktopBSD
- ✓ PCBSD



The NetBSD Project
"Of course it runs NetBSD"



PCBSD

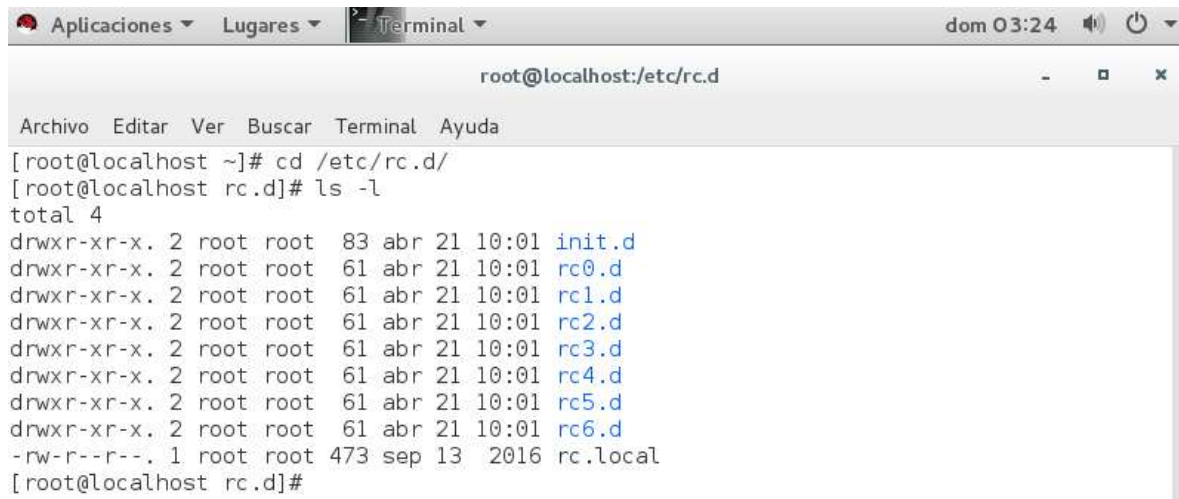
El fichero init.d

En este fichero se encuentran todos los scripts encargados de levantar cada uno de los servicios del servidor.

La ubicación de este fichero está localizada en:



[root@localhost ~]# **cd /etc/rc.d/** y luego dentro de la ruta ejecutamos el comando **ls -l**



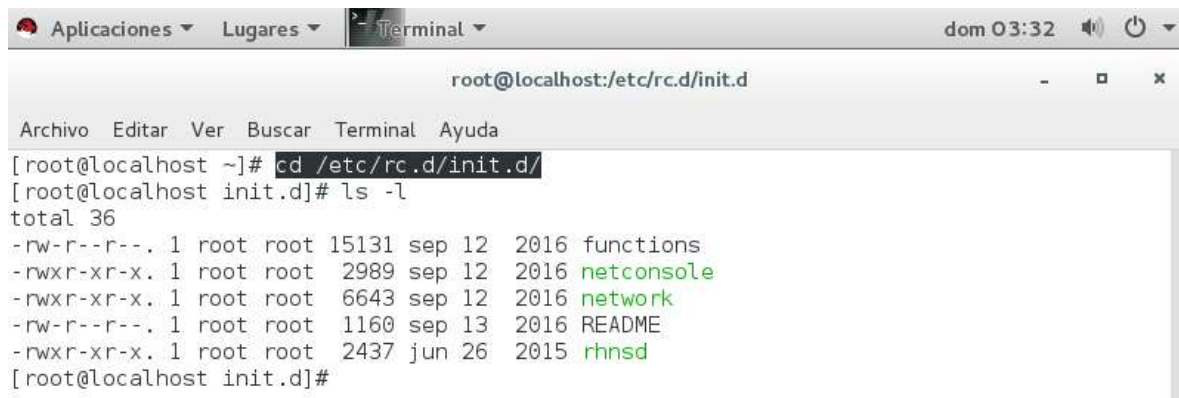
```
root@localhost:~# cd /etc/rc.d/
root@localhost rc.d# ls -l
total 4
drwxr-xr-x. 2 root root 83 abr 21 10:01 init.d
drwxr-xr-x. 2 root root 61 abr 21 10:01 rc0.d
drwxr-xr-x. 2 root root 61 abr 21 10:01 rc1.d
drwxr-xr-x. 2 root root 61 abr 21 10:01 rc2.d
drwxr-xr-x. 2 root root 61 abr 21 10:01 rc3.d
drwxr-xr-x. 2 root root 61 abr 21 10:01 rc4.d
drwxr-xr-x. 2 root root 61 abr 21 10:01 rc5.d
drwxr-xr-x. 2 root root 61 abr 21 10:01 rc6.d
-rw-r--r--. 1 root root 473 sep 13 2016 rc.local
root@localhost rc.d#
```

Figura 10: Resultado de la ejecución del comando `cd /etc/rc.d/`

Donde podemos observar los script del desde el **rc0.d** a el **rc6.d**

Algunos de los servicios que podemos encontrar en el fichero `init.d` son los referentes a: `functions`, `netconsole`, `network`, `rhnsd`, Servidor Web, `samba`, correo. `Dhcp`, `named` y `dbms`.

```
[root@localhost ~]# cd /etc/rc.d/init.d/
```



```
root@localhost:~# cd /etc/rc.d/init.d/
root@localhost init.d# ls -l
total 36
-rw-r--r--. 1 root root 15131 sep 12 2016 functions
-rwxr-xr-x. 1 root root 2989 sep 12 2016 netconsole
-rwxr-xr-x. 1 root root 6643 sep 12 2016 network
-rw-r--r--. 1 root root 1160 sep 13 2016 README
-rwxr-xr-x. 1 root root 2437 jun 26 2015 rhnsd
root@localhost init.d#
```

Figura 11: Resultado de la ejecución del comando `cd /etc/rc.d/init.d/`

El fichero `rcN.d`

rcN.d es un conjunto de directorios que representan cada uno de los niveles de ejecución del sistema operativo. Estos directorios a su vez contienen un conjunto de enlaces simbólicos a los scripts del directorio **/etc/rc.d/init.d**

La función que desempeñan estos directorios es organizar la manera en como los servicios de un servidor son levantados, como por ejemplo, imaginemos que tenemos instalado un servidor Web `apache`, y que lo tenemos configurado para que trabaje en los niveles de ejecución 3 y 5 , por ende deberíamos poder observar dichos enlaces simbólicos en las rutas.

- ✓ **/etc/rc.d/rc3.d**
- ✓ **/etc/rc.d/rc5.d**

Otra de las características de estos enlaces simbólicos es la sintaxis de sus propiedades. Esta sintaxis está conformada por 3 parámetros.

1. El estado del servicio, los cuales son representados con dos variables:

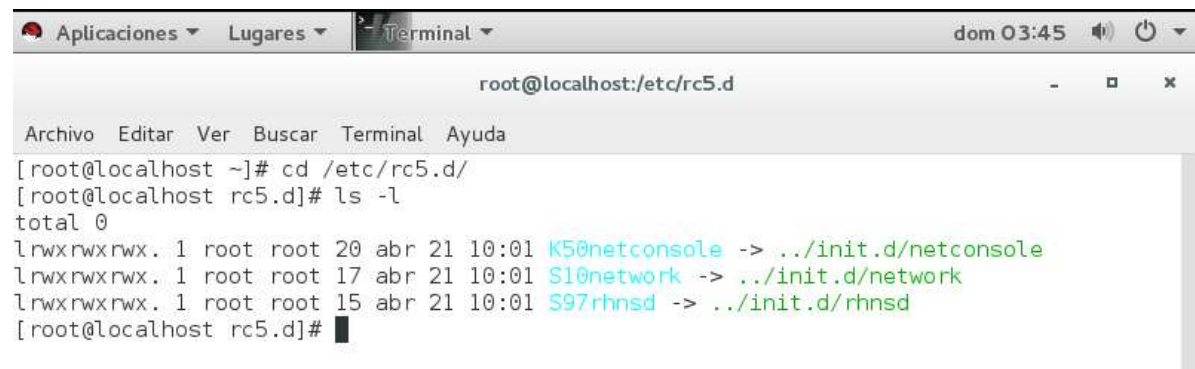
- ✓ La letra **K** Esta letra representa que el servicio está dado de baja
- ✓ La letra **S** Esta letra representa que el servicio está dado de alta

2. El orden en el que es arrancado el servicio. Este parámetro indica el orden en el que los servicios deben ser dados de alta o de baja

3. El nombre del servicio

Un ejemplo de esto lo podemos observar de la siguiente tabla, la cual la tomamos de la ruta **/etc/rc.d/rc5.d**, la que indica que los scripts dentro de esta carpeta se ejecutan en el nivel de ejecución 5

[root@localhost ~]# **cd /etc/rc5.d/** y luego ejecutamos [root@localhost rc5.d]# **ls -l** como indica la figura



```
root@localhost:~# cd /etc/rc5.d/
root@localhost rc5.d# ls -l
total 0
lrwxrwxrwx. 1 root root 20 abr 21 10:01 K50netconsole -> ../init.d/netconsole
lrwxrwxrwx. 1 root root 17 abr 21 10:01 S10network -> ../init.d/network
lrwxrwxrwx. 1 root root 15 abr 21 10:01 S97rhnsd -> ../init.d/rhnsd
root@localhost rc5.d#
```

Figura 12: Resultado de la ejecución del comando `cd /etc/rc5.d/`

lrwxrwxrwx. 1 root root 20 abr 21 10:01 K50netconsole -> ../init.d/netconsole

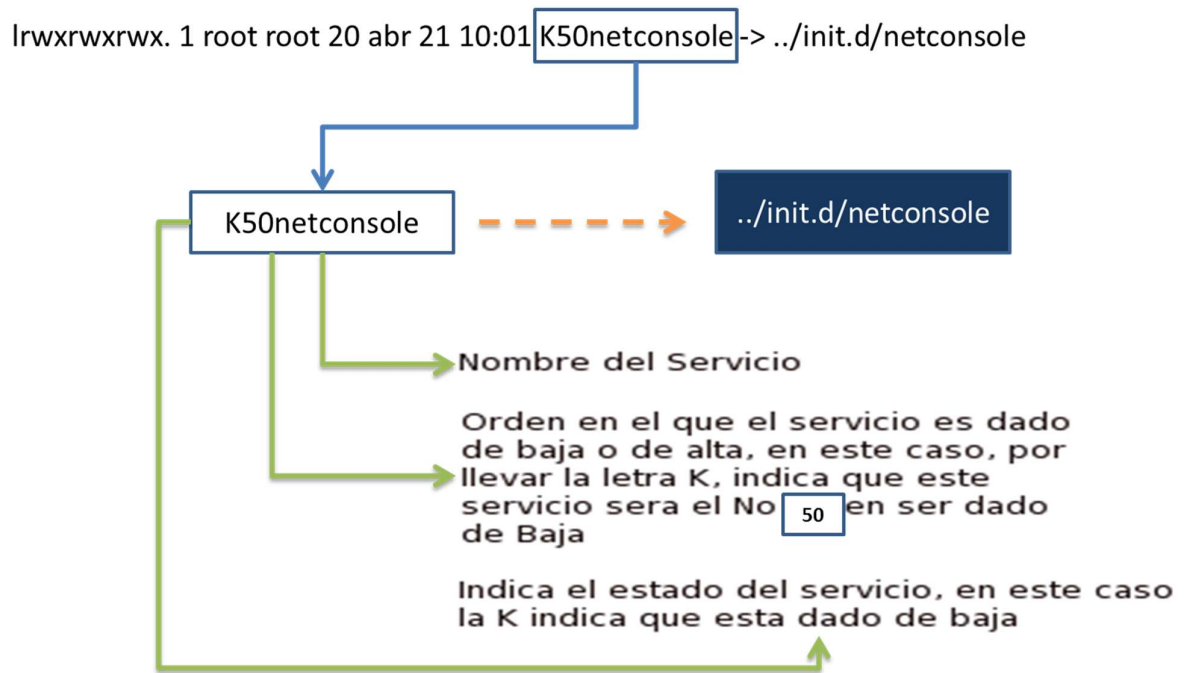


Figura 13: Orden del servicio dado de baja o dado de alta

Nivel 0 –Parada Del Sistema

El nivel 0 es usado para especificarle al sistema que debe apagarse, la forma en que este lo hace es a través del comando **halt**.

Al ejecutarse este comando se apagan todos los servicios que se encuentren activos



Figura 14: Resultado de la ejecución del comando halt

Nivel 1 o S Monousuario o Single User

El nivel 1 o también llamado nivel Single (S) solo puede ser iniciado por el administrador del sistema (root), por lo que ningún usuario podrá hacer eso de este nivel de ejecución.

En este nivel no se activan los servicios de Red, y tampoco se inician los procesos (daemons) de inicio por lo que permite reparar problemas o hacer pruebas al sistema.

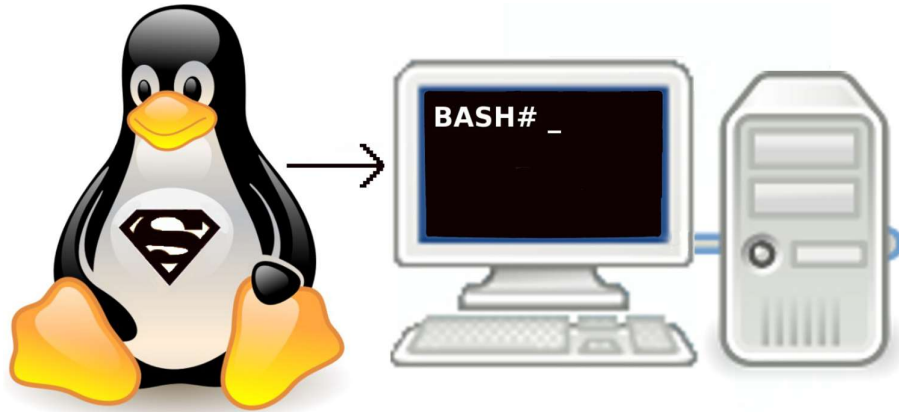


Figura 15: Monousuario o Single User

Nivel 2 Multiusuario sin Red

Este nivel está caracterizado por la capacidad de permitir que varios usuarios puedan entrar al sistema pero sin contar con soporte en red, esto quiere decir que no se puede contar con servidores como NFS o web.



Figura 16: Nivel 2 Multiusuario sin Red

Nivel 3 Multiusuario con Red

Este sistema está caracterizado por la capacidad de permitir a varios usuarios entrar al sistema, a diferencia del nivel de ejecución 2, este si cuenta con soporte de red.

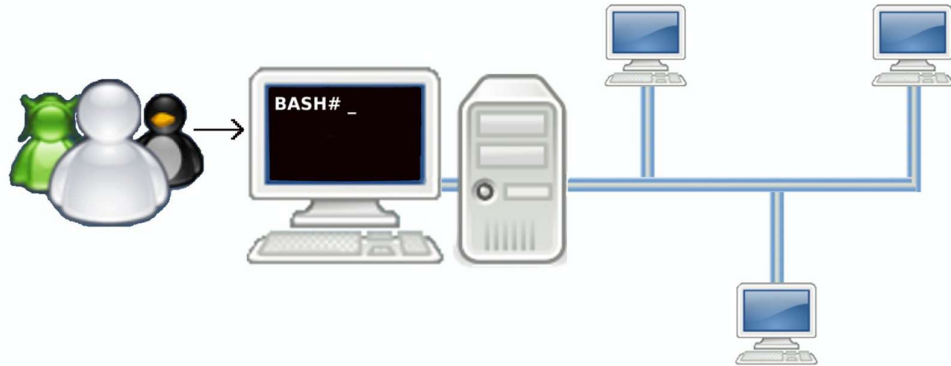


Figura 17: Nivel 3 Multiusuario con Red

Nivel 4. Sin Uso

Para la mayoría de las distribuciones Linux este nivel de ejecución no tiene asignada ninguna función, pero puede ser personalizado por el administrador para que cumpla con alguna función en especial.

Nivel 5. Multiusuario Grafico

Este nivel de ejecución es idéntico al nivel 3, la única diferencia es el alta de entornos gráficos como GNOME o KDE para la administración del sistema.



Figura 18: Nivel 5. Multiusuario Grafico

Nivel 6. Reinicio del Sistema



Figura 19: Nivel 6. Reinicio del Sistema

Pasar de niveles en niveles runlevel

Para este caso unas veces que estemos dentro del sistema operativo podemos movernos en los diferentes niveles del sistema operativo.



Recuerda que existen algunas distribuciones de Linux robustas que integran los todos los niveles de ejecución del 0 al 6.

Mediante el comando `[root@localhost ~]# who -r` obtenemos el resultado ``run-level' 5 2021-04-25 20:52`, el mismo que indica el nivel la fecha exacta y la hora exacta.

Una captura de pantalla de una ventana de terminal. La barra superior muestra 'Aplicaciones', 'Lugares' y 'Terminal'. El título de la ventana es 'Terminal' y la fecha/hora es 'dom 20:54'. El contenido de la terminal muestra el prompt `root@localhost:~` y el comando `who -r` ejecutado. El resultado es ``run-level' 5 2021-04-25 20:52`.

```
root@localhost:~  
Archivo  Editar  Ver   Buscar  Terminal  Ayuda  
[root@localhost ~]# who -r  
`run-level' 5 2021-04-25 20:52  
[root@localhost ~]#
```

Figura 20: Resultado de la ejecución del comando `who -r`

Para cambiar de niveles debemos de ejecutar en consola el siguiente comando:

`[root@localhost ~]# init 1` este comando nos permitirá cambiar de nivel 5 a el nivel 1, 1 o S **Monousuario o Single User**

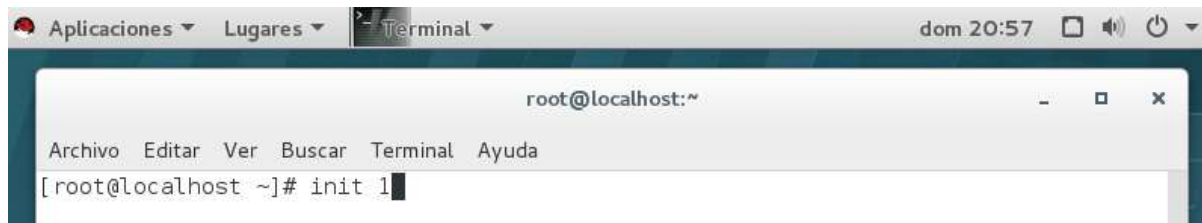


Figura 21: Resultado de la ejecución del comando init 1

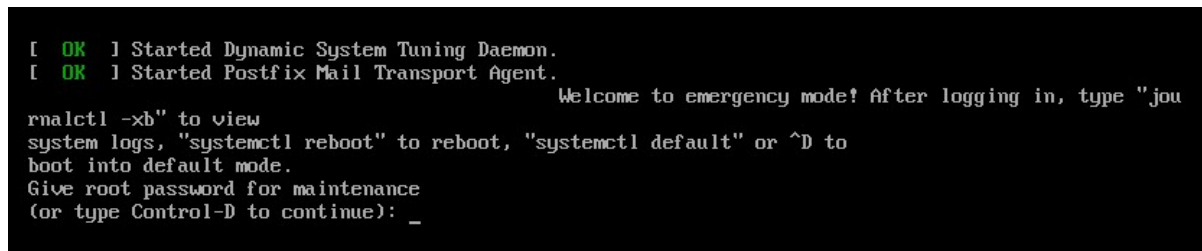


Figura 22: Resultado de la ejecución del comando init 1

En esta parte debemos de ingresar la clave del súper usuario root para continuar

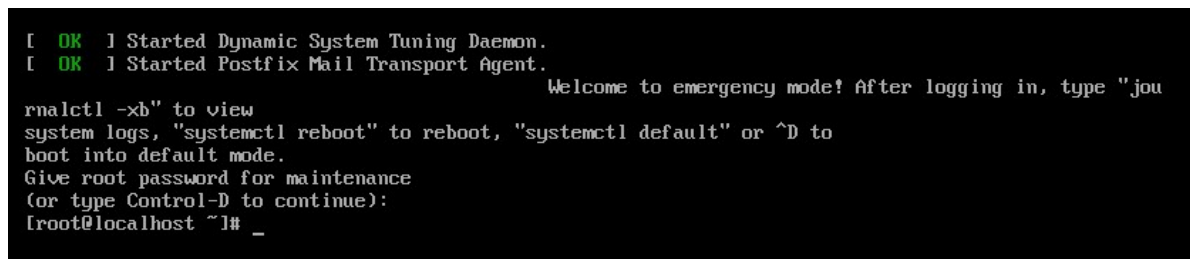


Figura 23: Resultado de la ejecución del comando init 1

Una vez ingresada la clave del súper usuario, nos aparecerá el intérprete para empezar a ingresar los cambios en caso que se desee hacer algún tipo de manteniendo del sistema.

Mediante el comando runlevel podemos identificar el nivel en que nos encontramos dentro del sistema operativo.

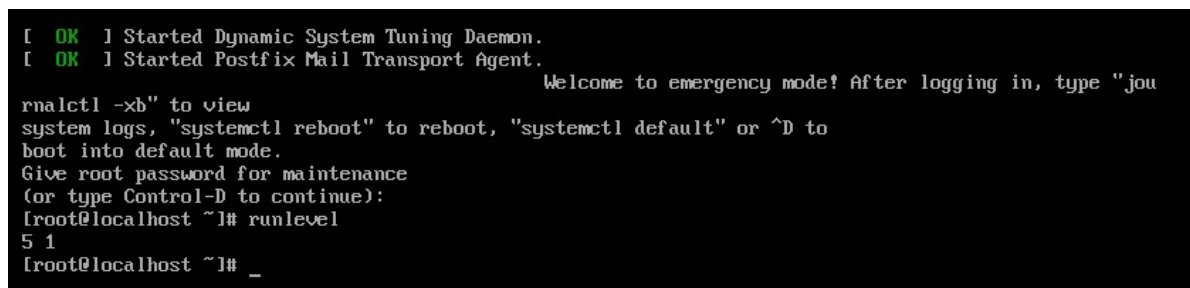


Figura 22: Resultado de la ejecución del comando runlevel en modo Single User

Lo que interpretamos de la imagen es que nos da como resultado de ejecutar el comando `runlevel 5 y 1`, lo que quiere decir que venimos de un **nivel 5** nivel grafico hemos pasado a un **nivel 1, Monousuario o Single User**.

Para pasar a otro nivel basta con ejecutar el comando `init 3` y automáticamente el sistema pasara a el nivel del sistema operativo

```
[ OK ] Started Dynamic System Tuning Daemon.  
[ OK ] Started Postfix Mail Transport Agent.  
Welcome to emergency mode! After logging in, type "journalctl -xb" to view  
system logs, "systemctl reboot" to reboot, "systemctl default" or ^D to  
boot into default mode.  
Give root password for maintenance  
(or type Control-D to continue):  
[root@localhost ~]# runlevel  
5 1  
[root@localhost ~]# init 3_
```

Figura 23: Resultado de la ejecución del comando `init 3` en modo Single User

```
Red Hat Enterprise Linux Server 7.3 (Maipo)  
Kernel 3.10.0-514.el7.x86_64 on an x86_64  
  
localhost login: root  
Password:
```

Figura 24: Resultado de la ejecución del comando `init 3`

Este nivel al contrario que el anterior nos va solicitar que ingresemos el nombre de usuario en este caso `root` y el password de acceso de `root`.

Para comprobar que estamos en el nivel 3 y que venimos del 1, solo bastaría con ingresar el comando `runlevel`, tal como se muestra en la imagen 1 3.

```
Red Hat Enterprise Linux Server 7.3 (Maipo)  
Kernel 3.10.0-514.el7.x86_64 on an x86_64  
  
localhost login: root  
Password:  
Last login: Sun Apr 25 21:04:38 on tty1  
[root@localhost ~]# runlevel  
1 3  
[root@localhost ~]#
```

Figura 25: Resultado de la ejecución del comando `runlevel` en nivel 3

Para volver al nivel 5, modo usuario grafico bastara con ingresar el comando `init 5`.

```
[root@localhost ~]# init 5_
```

Figura 26: Ejecución del comando `runlevel` en `init 5`

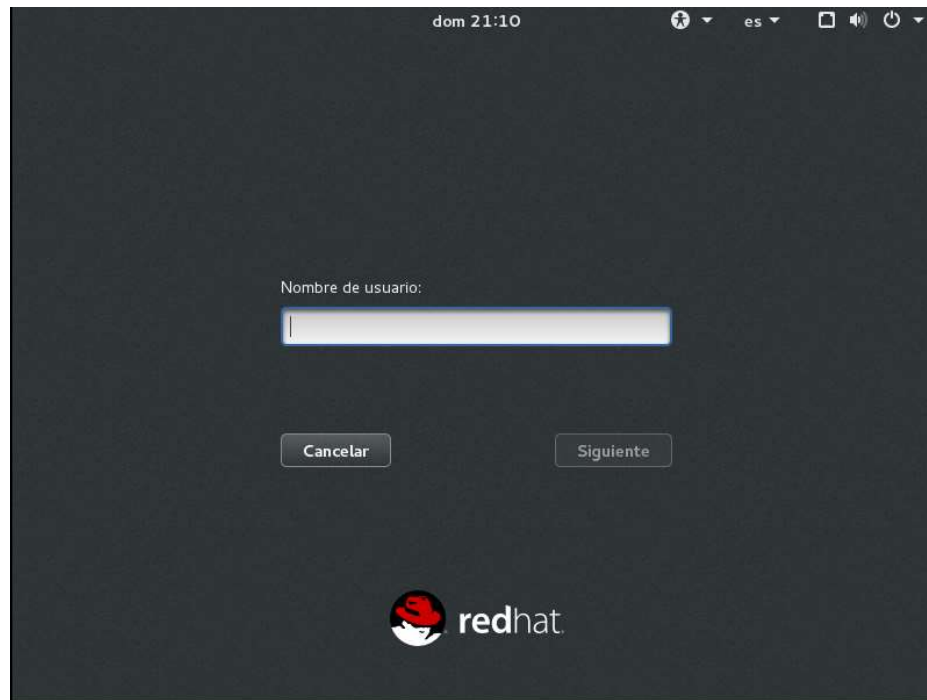


Figura 27: Resultado de la ejecución del comando `init 5`

Como podemos apreciar al invocar el comando `init 5`, nos trae la interfaz gráfica del sistema operativo.

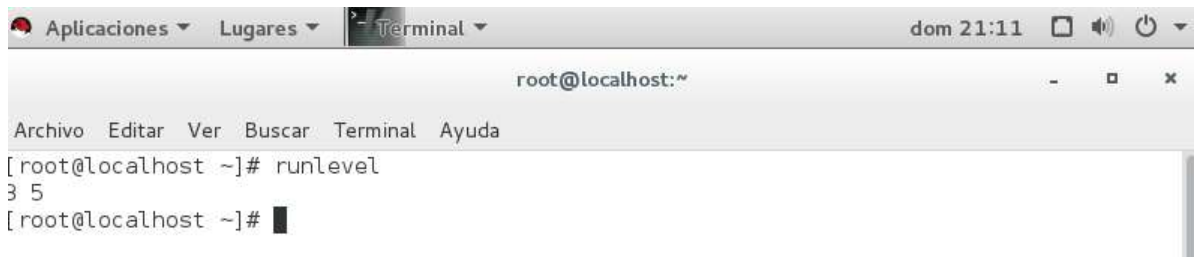


Figura 28: Resultado de la ejecución del comando `runlevel`

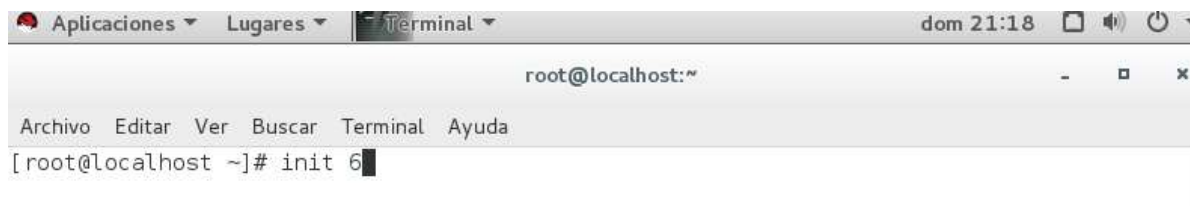
Como podemos observar en la imagen el resultado de la ejecución del comando `runlevel` nos dio `3 5`, lo que quiere decir que venimos del **nivel 3** y pasamos a el **nivel 5**. Mediante el comando `who -r` podemos observar mejor.



```
root@localhost:~  
Archivo Editar Ver Buscar Terminal Ayuda  
[root@localhost ~]# runlevel  
3 5  
[root@localhost ~]# who -r  
`run-level' 5 2021-04-25 21:09 último=3  
[root@localhost ~]#
```

Figura 29: Resultado de la ejecución del comando `who -r`

El comando `[root@localhost ~]# init 6`, permite realizar un reinicio del sistema operativo



```
root@localhost:~  
Archivo Editar Ver Buscar Terminal Ayuda  
[root@localhost ~]# init 6
```

Figura 30: Resultado de la ejecución del comando `init 6` reinicio del sistema

El comando `[root@localhost ~]# init 0`, permite realizar el apagado del sistema operativo

El GRUB del sistema operativo

GRUB es un programa de arranque de código abierto para sistemas Linux y UNIX. Los parámetros de arranque para el sistema se establecen en un archivo de configuración GRUB, que se carga al inicio. GRUB es flexible y permite realizar cambios en el momento del arranque, incluyendo la modificación de los parámetros del kernel e incluso la selección entre diferentes kernels que se pueden arrancar.

Para ahorrar espacio, así como reducir el tiempo de arranque, la imagen del kernel de Linux es un archivo comprimido que se extrae después de cargarse en la memoria. Durante el proceso de arranque, el gestor de arranque normalmente crea un sistema de archivos RAM temporal, conocido como *initramfs*. Este sistema de archivos contiene los controladores y módulos del núcleo necesarios que deben instalarse para admitir el sistema de archivos raíz real (que no está en la memoria principal). Una vez que el núcleo se ha iniciado y se instalan los controladores necesarios, el núcleo cambia el sistema de archivos raíz de la ubicación de RAM temporal a la ubicación del sistema de archivos raíz adecuada. Por último, Linux crea el proceso *systemd*, el proceso inicial en el sistema y, a continuación, inicia otros servicios (por ejemplo, un servidor web o una base de datos). En última instancia, el sistema presentará al usuario un mensaje de inicio de sesión.

Es importante tener en cuenta que el mecanismo de arranque no es independiente del gestor de arranque. Por lo tanto, hay versiones específicas del cargador de arranque para BIOS y UEFI, y el firmware debe saber también qué gestor de arranque específico se va a utilizar. El proceso de arranque para sistemas móviles es ligeramente diferente del de los PC tradicionales. Por ejemplo, aunque su núcleo está basado en Linux, Android no utiliza GRUB (gestor de arranque

por defecto de linux) y en su lugar lo deja a los proveedores para proporcionar cargadores de arranque. El cargador de arranque Android más común es LK (para "Little Kernel"). Los sistemas Android utilizan la misma imagen de núcleo comprimida que Linux, así como un sistema de archivos RAM inicial. Sin embargo, mientras que Linux descarta los *initramfs* una vez que se han cargado todos los controladores necesarios, Android mantiene *initramfs* como el sistema de archivos raíz para el dispositivo. Una vez que el núcleo se ha cargado y el sistema de archivos raíz montado, Android inicia el proceso de inicio y crea una serie de servicios antes de mostrar la pantalla de inicio.

Por último, los cargadores de arranque para la mayoría de los sistemas operativos, incluidos Windows, Linux y macOS, así como iOS y Android, proporcionan arranque en modo de recuperación o modo de usuario único para diagnosticar problemas de hardware, corregir sistemas de archivos dañados e incluso reinstalar el sistema operativo. Además de los errores de hardware, los sistemas informáticos pueden sufrir errores de software y un rendimiento deficiente del sistema operativo.

El grub de un sistema operativo Linux Red Hat

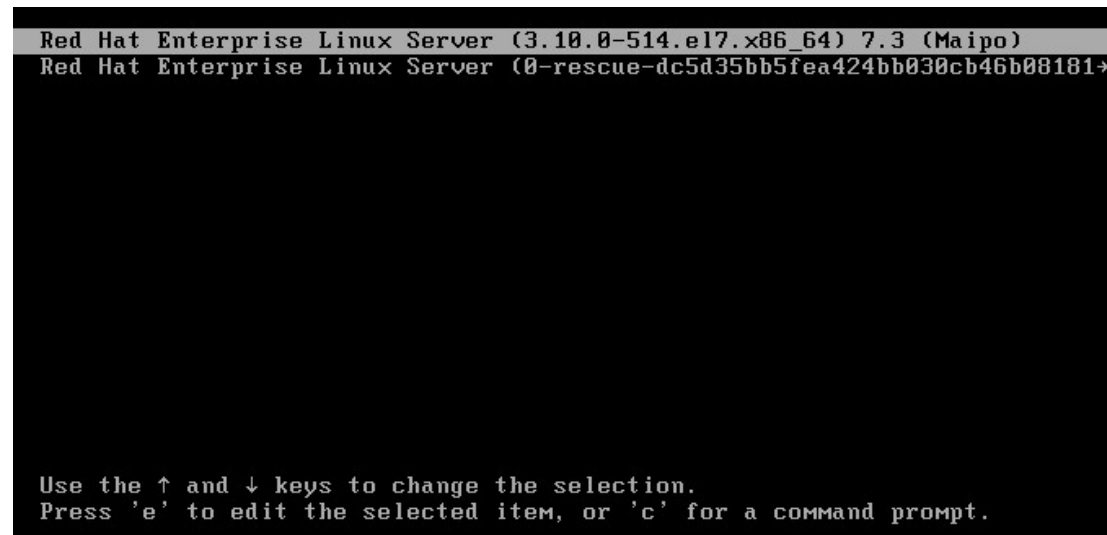


Figura 31: Acceso al GRUB del sistema operativo

Cabe indicar que la imagen muestra el GRUB de un sistema operativo robusto como es Red Hat, en este nivel se debe tener en cuenta procesos de hardening de sistemas operativos, debido a que si alguna persona con conocimientos de medios de sistemas operativos puede acceder y vulnerar el sistema como tan en su máximo nivel.

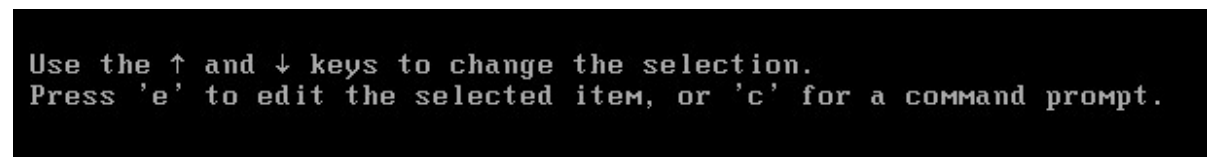





Figura 32: edición del GRUB del sistema operativo

Si nos detenemos y hacemos usos de la lectura:



Use the  keys to changes the selection press **e** to edit the selected item or **c** for a command prompt



Use las teclas direccionales para cambiar la selección presione **e** para editar el ítem seleccionado o **c** para un símbolo del sistema

Podemos proceder a editar el Grub del sistema operativo, si escogemos la opción e, podremos editar las siguientes opciones

```
setparams 'Red Hat Enterprise Linux Server (3.10.0-514.el7.x86_64) 7.3 (Maipo)\n',\n\nload_video\nset gfxpayload=keep\ninsmod gzio\ninsmod part_msdos\ninsmod xfs\nset root='hd0,msdos1'\nif [ x$feature_platform_search_hint = xy ] ; then\n    search --no-floppy --fs-uuid --set=root --hint-bios=hd0,msdos1 --hin\nt-efi=hd0,msdos1 --hint-baremetal=ahci0,msdos1 --hint='hd0,msdos1' eb7cd5f0-0\b86-48cf-9221-f13d85a4c77d\nelse\n    search --no-floppy --fs-uuid --set=root eb7cd5f0-0b86-48cf-9221-f13d\nd\n\nPress Ctrl-x to start, Ctrl-c for a command prompt or Escape to\ndiscard edits and return to the menu. Pressing Tab lists\npossible completions.
```

Figura 33: edición del GRUB del sistema operativo Linux Redhat 7.4

En este nivel podemos observar que no existe ningún método de seguridad aplicado en el proceso de despliegue del sistema operativo, por lo que sin conocer la clave de acceso del súper usuario podemos acceder al sistema haciendo uso de habilidades a ejecutar dentro de estas opciones.

Una vez dentro del sistema operativo vamos a analizar el grub y ver las opciones para encriptar el acceso al mismo el comando es el siguiente:

```
[root@localhost ~]# grub2-setpassword
```



```
root@localhost:~  
Archivo Editar Ver Buscar Terminal Ayuda  
[root@localhost ~]# grub2-setpassword  
Enter password:  
Confirm password:  
[root@localhost ~]#
```

Figura 34: hardening del grub2 de Linux Redhat

Como podemos apreciar se procedió a colocar un password de acceso a el grub2 del sistema operativo, lo que quiere decir que cuando algún usuario quiera editar deberá solicitarle un usuario y clave para poder acceder a las funciones del kernel del sistema operativo.

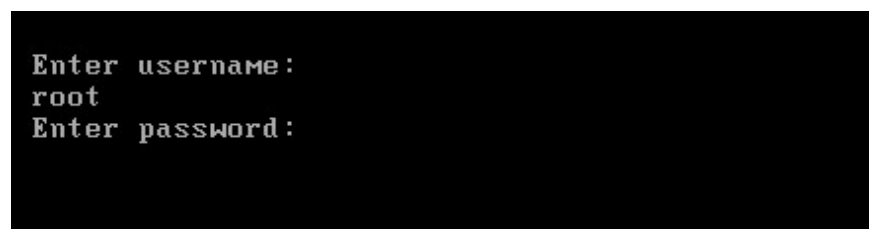
Luego de haber realizado los pasos anteriores debemos reiniciar el sistema operativo mediante el comando `[root@localhost ~]# init 6`



```
Red Hat Enterprise Linux Server (3.10.0-514.el7.x86_64) 7.3 (Maipo)  
Red Hat Enterprise Linux Server (0-rescue-dc5d35bb5fea424bb030cb46b08181)  
  
Use the ↑ and ↓ keys to change the selection.  
Press 'e' to edit the selected item, or 'c' for a command prompt.
```

Figura 35: Acceso al GRUB del sistema operativo cifrado

Procedemos a ingresar la letra **e** para editar y a continuación deberá de solicitar el usuario y la clave de acceso al **grub** y por ende al **kernel** del sistema operativo.



```
Enter username:  
root  
Enter password:
```

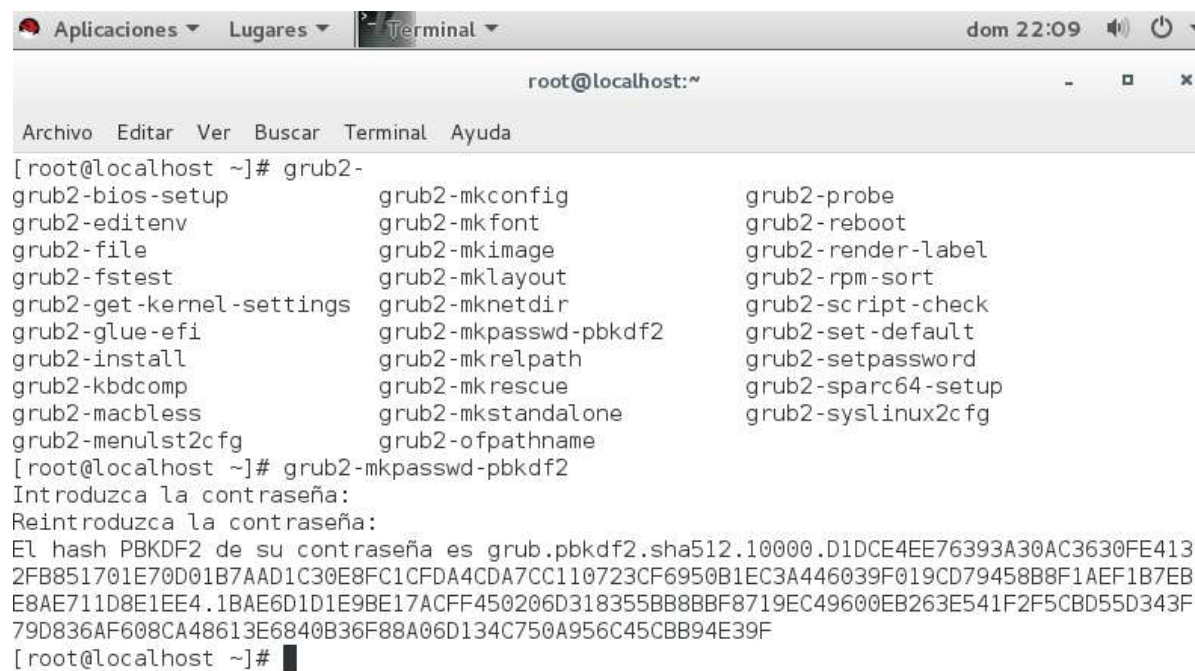
Figura 36: Acceso al GRUB del sistema operativo endurecido

Si las credenciales son correctas deberá de permitir realizar la manipulación del **grub** y opciones del kernel del sistema.

```
setparams 'Red Hat Enterprise Linux Server (3.10.0-514.el7.x86_64) 7.3 (Maipo)\n
\n
load_video
set gfxpayload=keep
insmod gzio
insmod part_msdos
insmod xfs
set root='hd0,msdos1'
if [ x${feature_platform_search_hint} = xy ]; then
  search --no-floppy --fs-uuid --set=root --hint-bios=hd0,msdos1 --hin\
t-efi=hd0,msdos1 --hint-baremetal=ahci0,msdos1 --hint='hd0,msdos1' eb7cd5f0-0\
b86-48cf-9221-f13d85a4c77d
else
  search --no-floppy --fs-uuid --set=root eb7cd5f0-0b86-48cf-9221-f13d\
\n
Press Ctrl-x to start, Ctrl-c for a command prompt or Escape to
discard edits and return to the menu. Pressing Tab lists
possible completions.
```

Figura 37: edición del GRUB del sistema operativo endurecido mediante credenciales establecidas


Proceso de encriptación del grub del sistema operativo



```
root@localhost:~\n
\n
Archivo Editar Ver Buscar Terminal Ayuda\n
[ root@localhost ~ ]# grub2-\n
grub2-bios-setup          grub2-mkconfig          grub2-probe\n
grub2-editenv             grub2-mkfont            grub2-reboot\n
grub2-file                grub2-mkimage           grub2-render-label\n
grub2-fstest              grub2-mklayout          grub2-rpm-sort\n
grub2-get-kernel-settings grub2-mknetdir           grub2-script-check\n
grub2-glue-efi            grub2-mkpasswd-pbkdf2   grub2-set-default\n
grub2-install             grub2-mkrelpath         grub2-setpassword\n
grub2-kbdcomp             grub2-mkrescue          grub2-sparc64-setup\n
grub2-macbless            grub2-mkstandalone      grub2-syslinux2cfg\n
grub2-menulst2cfg         grub2-ofpathname\n
[ root@localhost ~ ]# grub2-mkpasswd-pbkdf2\n
Introduzca la contraseña:\n
Reintroduzca la contraseña:\n
El hash PBKDF2 de su contraseña es grub.pbkdf2.sha512.10000.D1DCE4EE76393A30AC3630FE413\n
2FB851701E70D01B7AAD1C30E8FC1CFDA4CDA7CC110723CF6950B1EC3A446039F019CD79458B8F1AEF1B7EB\n
E8AE711D8E1EE4.1BAE6D1D1E9BE17ACFF450206D318355BB8BBF8719EC49600EB263E541F2F5CBD55D343F\n
79D836AF608CA48613E6840B36F88A06D134C750A956C45CBB94E39F\n
[ root@localhost ~ ]#
```

Figura 38: opciones de configuración del GRUB2

Aquí te dejo todas las opciones para que realices prácticas sobre las funciones del **GRUB**, de un sistema operativo robusto para servidores.



Sabías que Red Hat trabaja en por defecto en **nivel de ejecución 5**, mientras que Ubuntu trabaia en **nivel 2**.

El GRUB del sistema operativo Ubuntu.

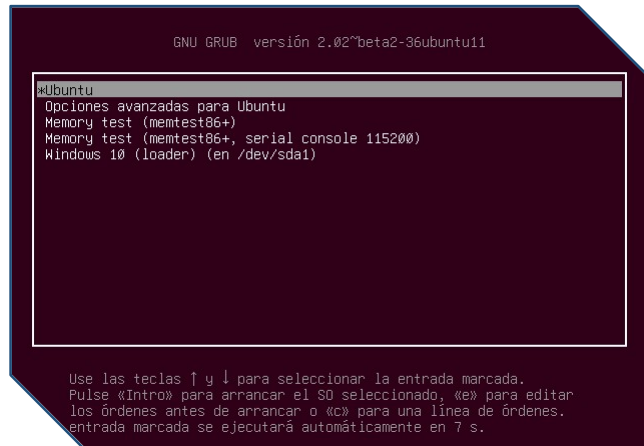


Figura 19: GRUB, mostrando arranque doble con Ubuntu y Windows

GNU GRUB (*GNU* → **GR**and **Un**ified *Bootloader*) es un cargador de arranque múltiple, desarrollado por el proyecto GNU. GRUB es la implementación de referencia de la Free Software Foundation para la especificación de arranque múltiple, que brinda al usuario la opción de arrancar uno de los múltiples sistemas operativos instalados en una computadora o seleccionar una configuración de núcleo específico de un

sistema operativo en particular. GRUB se usa a menudo como un selector de

sistema operativo en el inicio para computadoras de arranque dual, por ejemplo, donde un sistema es Linux y el otro Windows.

Características:

GRUB es un cargador de arranque que cumple la Especificación Multiboot. De esta forma puede arrancar de forma uniforme los kernels que cumplen la Especificación Multiboot.

Está diseñado para cumplir los siguientes objetivos, listados por orden de importancia:

- Funciones básicas tienen que ser sencillas para los usuarios finales.
- Amplia funcionalidad para apoyar a los expertos y diseñadores del kernel.
- Compatibilidad con versiones anteriores para arrancar FreeBSD, NetBSD, OpenBSD y Linux. Los núcleos compatibles como DOS, Windows NT y OS/2, son arrancables mediante una función de carga en cadena.

Características adicionales que soporta:

- Reconoce múltiples formatos ejecutables.
- Soporta núcleos que no cumplen especificación multiboot.
- Admite comandos de configuración y puede cargar una configuración preestablecida.
- Provee interfaz de menú.
- Tiene una interfaz de línea de comandos flexible.
- Admite muchos sistemas de ficheros (BtrFS, ext2/ext3/ext4, FAT12/FAT16/FAT32, exFAT, HFS, HFS+ ISO9660, nilfs2, NTFS, ReiserFS, ZFS, ROMFS, JFS, XFS, ...).
- Descompresión automática.
- Acceso a datos ubicados en cualquier dispositivo instalado.
- Independiente de la geometría de la unidad.
- Detecta toda la RAM instalada.
- Admite direccionamiento de bloque lógico (LBA).
- Arranque de red.
- Terminales remotos para permitir el control desde una estación remota.

Otros gestores de arranque:

Adicional a GRUB, existen otros gestores de arranque, entre los que se puede destacar:

Boot Configuration Data:

Boot Configuration Data (BCD) es una base de datos independiente del firmware para los datos de configuración en el momento del arranque. Lo usa el nuevo Administrador de arranque de Windows (*Windows Boot Manager*) de Microsoft a partir de Microsoft Windows Vista y reemplaza el archivo boot.ini que usaba NTLDR (*NT Loader*), el cual era el gestor

de arranque por defecto hasta Microsoft Windows XP.



Figura 20: Windows Boot Manager

El Windows BOOTMGR (*Boot Manager*) eventualmente ejecutará el archivo winload.exe, el gestor de arranque del sistema operativo para cargar el ejecutable del núcleo del sistema operativo (ntoskrnl.exe) y los controladores de dispositivos principales. En sistemas UEFI el archivo es llamado winload.efi.

BootX/iBoot:

BootX es un cargador de arranque basado en software diseñado y desarrollado por Apple Inc. para su uso en la gama de ordenadores Macintosh de la empresa. BootX se usa para preparar la computadora para su uso, cargando todos los controladores de dispositivo requeridos y luego iniciando Mac OS X al arrancar el núcleo en todos los PowerPC Macintosh que ejecutan el sistema operativo Mac OS X 10.2 o versiones posteriores.



Figura 41: BootX de Apple

Por su parte iBoot es el gestor de arranque de la etapa 2 para todos los productos Apple. Reemplaza al antiguo cargador de arranque, BootX. En comparación con su predecesor, iBoot mejora la autenticación realizada en la cadena de arranque.

Para macOS, el proceso de arranque comienza con la ejecución del código almacenado en la ROM de arranque seguro UEFI (primera etapa). Boot ROM tiene dos responsabilidades principales: inicializar el hardware del sistema (el componente POST) y seleccionar un sistema operativo para ejecutar (el componente UEFI).

Interfaz de usuario del sistema operativo.

En el campo del diseño industrial de la interacción humano-computadora, una interfaz de usuario (UI) es el espacio donde ocurren las interacciones entre humanos y máquinas. El objetivo de esta interacción es permitir la operación y el control efectivos de la máquina desde el extremo humano, mientras que la máquina retroalimenta simultáneamente información que ayuda en el proceso de toma de decisiones de los operadores. Ejemplos de este amplio concepto de interfaces de usuario incluyen los aspectos interactivos de los sistemas operativos de computadora, herramientas manuales, controles de operador de maquinaria pesada y controles de proceso. Las consideraciones de diseño aplicables al crear interfaces de usuario están relacionadas o involucran disciplinas como la ergonomía y la psicología.

En el campo de los sistemas operativos usualmente se cuenta con la interfaz de línea de comandos (CLI → *Command Line Interface*) y la interfaz gráfica de usuario (GUI → *Graphic User Interface*).

Interfaz de línea de comandos:

Una interfaz de línea de comandos (CLI) procesa los comandos de un programa de computadora en forma de líneas de texto. El programa que maneja la interfaz se llama intérprete de línea de comandos o procesador de línea de comandos. Los sistemas operativos implementan una interfaz de línea de comandos en un shell para el acceso interactivo a las funciones o servicios del sistema operativo.

Entre las interfaces de línea de comando más utilizadas actualmente tenemos:

- Microsoft PowerShell: Interfaz de línea de comando usado actualmente en Microsoft Windows, macOS X, y entornos Linux.
- Bash: es una interfaz Shell de UNIX e intérprete de comandos, usado ampliamente en entornos UNIX-Like (Linux), macOS y actualmente en Microsoft Windows a través de WSL. En las diferentes versiones de Linux en el entorno gráfico se utiliza un emulador del terminal.
- CMD: también llamado Command Prompt, es el intérprete de comandos que usa actualmente Microsoft Windows, aunque es usado por otros sistemas operativos antiguos, el CMD comparte muchas de sus funcionalidades con el antiguo COMMAND.COM, que era el intérprete de comandos del ya extinto MS-DOS.

Interfaz Gráfica de Usuarios:

La interfaz gráfica de usuarios permite al usuario interactuar con dispositivos electrónicos de una forma más intuitiva a través de iconos gráficos e indicadores de audio. Las acciones en una GUI son usualmente llevadas a través de la manipulación directa de los gráficos.

Yendo más allá de las GUI para computadores, las GUI son utilizadas actualmente para manipular dispositivos como teléfonos inteligentes, consolas de videojuegos, reproductores MP3, entre otros.

Existen varios GUI entre los sistemas operativos existentes:



Microsoft Windows



Apple macOS



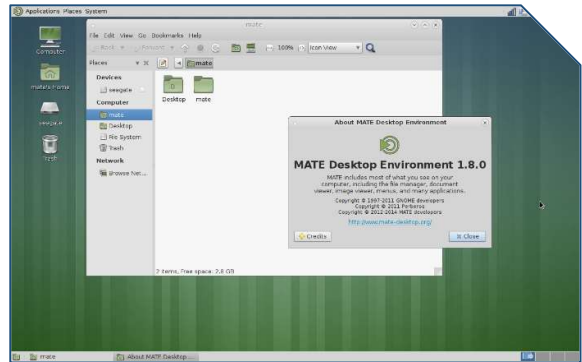
Linux

En Linux tenemos tantos GUI como sabores puedan existir, entre los más utilizados destacamos:

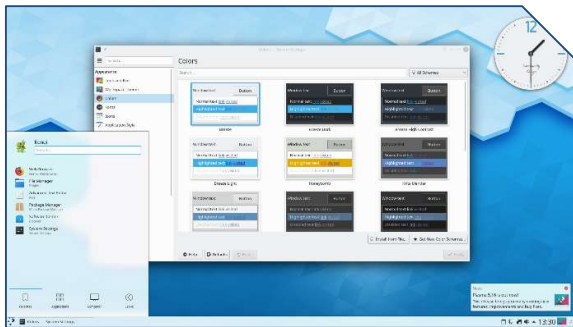
Gnome 3



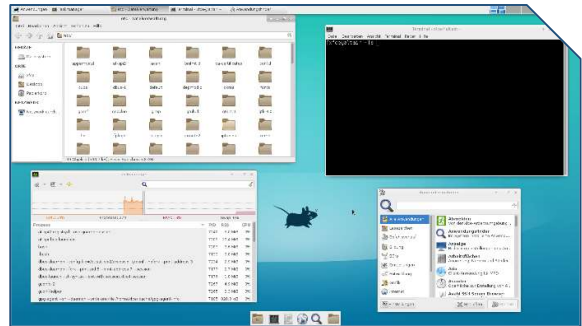
Mate



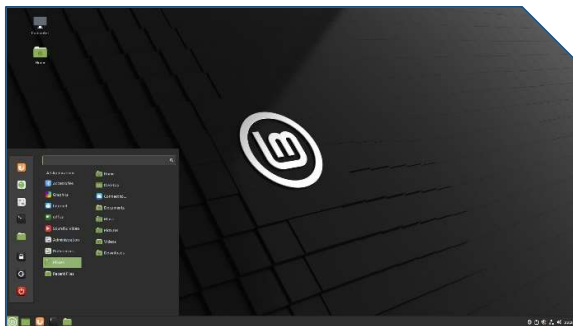
KDE 5



XFCE



Cinnamon



LXDE





Tema 2: Estructura de los sistemas operativos

Se deberá elegir un método para estructurar las funcionalidades que se proveerán. Actualmente los sistemas operativos se encuentran divididos por jerarquías con funciones bien definidas. Se mencionarán algunas formas de estructurar el sistema a continuación.

Estructura simple

La estructura simple está orientada a sistemas operativos pequeños, simples y a su vez limitados. Por ejemplo, MS-DOS entregaba máximas funcionalidades en un tamaño reducido, no poseía una división cuidadosa de sus módulos. Adicionalmente dicho sistema operativo entregaba acceso directo a rutinas que podían utilizar el hardware, por lo cual no se considera un sistema operativo con protección de sus procesos. En el caso del Unix original, el núcleo a través de las llamadas al sistema provee las funcionalidades necesarias para acceder a los recursos.

Estructura en niveles

En este tipo de estructuras se utiliza un método de diseño arriba – abajo, el sistema resultante corresponderá a un sistema por niveles donde la estructura jerárquica se determinará de acuerdo con la complejidad de las funciones de cada nivel.

Las ventajas de utilizar esta estructura radican en la independencia que se conseguirá entre los niveles, ya que cada uno se encargará de una tarea específica que le entregará servicios a otro nivel. Se debe preocupar mantener las mismas funcionalidades que se entregan a otras capas, no importando como se cambie esto internamente. Esto proporciona facilidad en la construcción, mantención y depuración del sistema operativo.

Se debe tener especial cuidado en la definición apropiada de los diferentes niveles, donde esto debe hacerse de forma correcta para lograr la independencia anteriormente mencionada.

Además, se debe considerar que ciertas capas podrán depender de otras para operar. Una desventaja es que al introducir niveles la operación total podría resultar un poco más lenta, ya que se deben utilizar interfaces entre las diferentes capas del sistema.

A continuación, se muestra un ejemplo de los posibles niveles de jerarquía para un sistema operativo. Notar los niveles del 1 al 4 no corresponden directamente a funciones del sistema operativo, estas son realizadas por hardware. También observar que las capas superiores requerirán servicios de capas inferiores como es el caso del nivel de directorios que requiere servicios de la capa sistema de archivos y esta a su vez de la capa de almacenamiento secundario.

- Circuitos electrónicos: registros, puertas, buses.
- Instrucciones: evaluación de la pila, microprogramas, vectores de datos.
- Procedimientos: pila de llamadas, visualización.
- Interrupciones: manejo de interrupciones del hardware.
- Procesos primitivos: semáforos, colas de procesos.
- Almacenamiento secundario: bloques de datos.
- Memoria virtual: paginación.
- Comunicaciones: tuberías.

- Sistema de archivos: almacenamiento en disco duro u otro medio.
- Dispositivos: impresoras, pantallas, teclados.
- Directorios: árbol de directorios.
- Procesos de usuario: programas en ejecución.
- Shell: intérprete de comandos.

Microkernels

Un sistema operativo que está organizado como micronúcleo entrega solo las tareas básicas, como: planificación de procesos, gestor de memoria y comunicaciones. Otras tareas son realizadas por programas del sistema operativo y el núcleo es utilizado como un intermediario para la comunicación entre el usuario y los programas del sistema operativo que ofrecen los servicios.

Los programas nuevos para el sistema operativo son añadidos al espacio del usuario, se ejecutan en modo usuario y no como modo sistema. El núcleo entonces se encarga de realizar las llamadas al sistema a través de mensajes hacia los servicios correspondientes que entregan las funcionalidades solicitadas.

Su ventaja es que, al incorporar las mínimas funcionalidades, son más estables. Sin embargo, la principal desventaja en este tipo de núcleos es que son ineficientes al tener que realizar muchos cambios de contexto para ir a los servicios prestados. Minix es un ejemplo de este tipo de sistema operativo.

Módulos

En este caso el sistema operativo está compuesto por módulos, donde lo fundamental se encuentra en el núcleo en sí, pero otras funcionalidades son entregadas como módulos del núcleo. Ambos, tanto el núcleo como los módulos corren en modo sistema. Esto permite que componentes sean cargados dinámicamente al núcleo, evitando tener que disponer del soporte para todos los dispositivos o funcionalidades permanentemente cargados en memoria principal. En Linux esto se puede realizar mediante el uso de las instrucciones `lsmod`, `modprobe` y `rmmod`.

Algunos ejemplos de módulos que pueden existir son controladores de disco, controladores de tarjetas de red o el soporte para IPv6. Es importante mencionar que el soporte necesario para que la máquina pueda ser arrancada, en estricto rigor para que el disco duro que contiene el sistema raíz del sistema operativo sea abierto, no puede ir como módulo del núcleo. Lo anterior ya que los módulos se cargan cuando el sistema está iniciando, una vez que ya se montó el sistema de archivos. Ejemplos de estos sistemas operativos son Unix modernos, Solaris, Linux y Mac OSX.

Clasificación de los sistemas operativos

En este punto se realizará una mención por los distintos tipos de Sistemas Operativos, dependiendo del número de aplicaciones que ejecutan, del número de usuarios que interactúan, del número de procesadores que manejan o dependiendo de la estructura. A continuación, se enumeran los sistemas mencionados:

Según la utilización de los recursos:

Si un sistema operativo se ejecuta en una computadora con un solo procesador se clasifican en sistemas monoprogramados y multiprogramados. Sin embargo, si tiene varios procesadores se habla de sistemas multiprocesamiento.

- Sistema monoprogramado. Sólo se permite la ejecución de un programa en el sistema, por lo que el programa se carga en memoria y permanece en ella hasta su finalización.
- Sistema multiprogramado o multitarea. Estos sistemas pueden ejecutar simultánea varias tareas y de varios usuarios. El sistema operativo mantiene varios trabajos en la memoria, y mientras uno está a la espera de realizar alguna tarea, por ejemplo, la terminación de una operación de entrada/salida (E/S), ya sea el presionar una tecla u otra tarea, el sistema operativo selecciona otro trabajo para que la CPU lo ejecute y esta última no quede vacía de trabajo mientras estos esperan su ejecución. Hoy en día todos los sistemas operativos son multitarea.
- Sistema multiprocesador. Estos sistemas son aquellos que pueden manejar varios procesadores, con lo que se alcanza un mayor rendimiento ya que al tener más procesadores podemos realizar más tareas al mismo tiempo, aunque hay que decir que este aumento no es proporcional al aumento del número de procesadores. A este tipo de sistema también se le llama computadoras paralelas o multicomputadoras, ya que cada procesador tiene una copia del sistema operativo, y éstas se comunican entre sí para el manejo de las tareas, repartiéndose las mismas entre los procesadores, es decir, una tarea es ejecutada en proporción por todos los procesadores.

Según la interactividad:

En este caso se habla de los sistemas dependiendo del tipo de trabajo y los servicios que se presta a los usuarios se habla de sistemas de procesos por lotes, sistemas de tiempo compartido y sistemas de tiempo real.

- Sistemas de procesamiento por lotes (Batch). En este tipo de sistemas no existe intervención del usuario durante la ejecución de los trabajos. Cada trabajo consiste en una relación de pasos secuenciales, que al juntarse forman un lote. Estos procesamientos son largos y no tienen límite de tiempo en su ejecución.
- Sistemas de tiempo compartido. En este tipo de sistemas los procesos se ejecutan de manera simultánea mientras la CPU conmuta entre ellos, de manera que el usuario es ajeno a la misma. En estos sistemas los usuarios sí pueden interactuar con los programas durante su ejecución.
- Sistemas de tiempo real. Se caracterizan porque maximizan el tiempo en el que se manda la información, por ello se utilizan para procesos delicados, como por ejemplo la ejecución por parte del brazo de un robot en una planta de ensamblaje de vehículos. Dicho robot debe obtener la información de ejecución dentro de unos márgenes estrictos de tiempo, ni antes ni después.

Según el número de usuarios:

En este caso se habla de los sistemas dependiendo del tipo de trabajo y los servicios que se presta a los usuarios se habla de sistemas de procesos por lotes, sistemas de tiempo compartido y sistemas de tiempo real.

- Sistema operativo de computadora personal. Se puede también denominar sistema monousuario. Atiende al número de usuarios que utilizan dicho sistema al mismo tiempo. En el caso de los monousuario hasta que la persona que está utilizando el equipo no termina, éste no puede ser utilizado por otra. El sistema operativo de computadora personal tiende a ser monousuario, ya que es un sistema enfocado a maximizar la comodidad del usuario y a responder con rapidez a las necesidades de éste. Como ejemplo actual de este sistema podemos mencionar a Windows 7, que, excepto en algunas versiones mediante la herramienta de terminal server se puede convertir en multiusuario, la mayoría es monousuario.
- Sistema operativo servidor o multiusuario. Al contrario que el anterior, en este sistema concurren más de un usuario de forma simultánea desde diferentes ubicaciones. Por ejemplo, el caso de los servidores web, los cuales sirven las páginas alojadas en los mismos a través de internet a todo aquel que las visite. Entre los sistemas operativos de servidor encontramos a UNIX, Windows Server y Linux.

Según el número de computadoras:

Dependiendo del número de computadoras que formen parte del sistema, se puede hablar de sistemas centralizados o sistemas distribuidos.

- Sistemas centralizados. Este tipo de sistemas es aquel que utiliza los recursos de una sola computadora, como es la memoria, CPU, discos, periféricos. Este tipo de sistema planteaba un problema, y es que cuando se interconectaban ordenadores a uno principal, cuando este último requería un cambio de hardware para aumentar su capacidad, ello era más costoso que añadir nuevos computadores.
- Sistemas distribuidos. Como lo define Tanenbaum, “Un sistema distribuido es una colección de computadoras independientes que aparecen ante los usuarios del sistema como una única computadora”. En este tipo de sistemas se utiliza al mismo tiempo los recursos de los distintos equipos, ya bien sea el hardware o el software de los mismos, por lo que las computadoras se comunican entre sí a través de distintos medios de comunicación, como pueden ser líneas de alta velocidad o líneas telefónicas. Este tipo de sistemas puede abarcar desde unas pocas computadoras hasta miles o millones de ellas, enlazadas a través de internet. Estos recursos que son compartidos son manejados por un gestor de recursos, que es un software que se encarga de controlar dichos recursos.

Según la estructura del sistema operativo:

En este caso se encuentra los distintos sistemas dependiendo del diseño interno de los mismos. Entre ellos encontramos los sistemas monolíticos, los sistemas en capas, las máquinas virtuales y el sistema cliente-servidor.

- Sistemas monolíticos. Son sistemas pequeños, sencillos y limitados. No tienen una estructura definida, sino que todos sus componentes se encuentran agrupados en un único programa. Cada conjunto de procedimientos puede invocar a cualquiera de los otros procedimientos. Un ejemplo de este tipo de sistemas es el sistema MS-Dos.
- Sistema por capas. El sistema se organiza como una jerarquía de capas, donde cada capa ofrece una interfaz a la capa superior, y sólo utiliza los servicios que ofrece la capa

inferior. Cada capa tiene funciones específicas, así cada capa se encarga de una parte del sistema operativo.

- Sistema de máquina virtual. Se separa las dos funciones que proporciona un sistema de tiempo compartido o multiusuario, como son la multiprogramación y la abstracción del hardware.
- Sistema cliente-servidor. El sistema consiste en un conjunto de módulos autónomos, los cuales ponen a disposición de los demás una serie de servicios o competencias. Es decir, cada uno de los módulos actúan como servidores que atienden las peticiones de otros módulos que actuarían como clientes. Este tipo de sistema es muy apropiado para los sistemas distribuidos.

Llamadas al sistema

Una llamada al sistema es la forma programática en la que un programa de ordenador solicita un servicio desde el núcleo del sistema operativo en el que se ejecuta. Esto puede incluir servicios relacionados con el hardware (por ejemplo, el acceso a una unidad de disco duro), la creación y ejecución de nuevos procesos, y la comunicación con integrales servicios del núcleo, tales como la planificación de procesos. Las llamadas al sistema proporcionan una interfaz esencial entre un proceso y el sistema operativo.

En la mayoría de los sistemas, las llamadas al sistema sólo se pueden hacer a partir del espacio de usuario procesos, mientras que, en algunos sistemas, OS / 360 y sucesores, por ejemplo, el código del sistema privilegiada también emite las llamadas al sistema.

En general, los sistemas proporcionan una biblioteca o API que se encuentra entre los programas normales y el sistema operativo. En sistemas Unix, la API es generalmente parte de una implementación de la biblioteca de C (libc), tales como glibc, que proporciona funciones de contenedor (wrapper, es una subrutina o función en una biblioteca de software o un programa de computadora cuyo propósito principal es llamar a una segunda subrutina) para las llamadas al sistema, a menudo nombrado del mismo modo como la llamada del sistema los invocan.

En Windows NT, estas API son parte de la API nativa, en la biblioteca **ntdll.dll**; esto es una API no documentada utilizada por las implementaciones de la regularidad de la API de Windows y directamente utilizados por algunos programas de sistema en Windows. Las funciones de la biblioteca contenedor (wrapper) exponen a una función ordinaria de convención de llamada (una subrutina llama a nivel ensamblador) para el uso de la llamada al sistema, así como hacer la llamada al sistema más modular.

En este caso, la función principal de contenedor (wrapper) es colocar todos los argumentos que se pasan a la llamada al sistema en los correspondientes registros del procesador (y tal vez en la pila de llamadas también), y además del establecimiento de un número de llamada de sistema único del kernel hacia la llamada. De esta manera la biblioteca, que existe entre el sistema operativo y la aplicación, aumenta la portabilidad.

En sistemas operativos Unix, similares a Unix, POSIX, las llamadas populares al sistema son **open, read, write, close, wait, exec, fork, exit, y kill**. Muchos sistemas operativos modernos tienen cientos de llamadas al sistema. Por ejemplo, Linux y OpenBSD tienen cada uno más de 300

llamadas diferentes, NetBSD tiene cerca de 500, FreeBSD tiene más de 500, Windows 7 cuenta con cerca de 700.

La implementación de las llamadas al sistema requiere una transferencia de control desde el espacio de usuario al espacio del núcleo, lo que implica algún tipo de función de una arquitectura específica. Una forma típica de implementar esto es utilizar una interrupción de software o excepción (**trap**, también conocida como excepción o falla, es típicamente un tipo de interrupción síncrona causada por una condición excepcional, por ejemplo, punto de interrupción, división por cero, acceso a memoria inválido). Las interrupciones transfieren el control al kernel del sistema operativo, de manera que el software simplemente necesita establecer algún registro con el número de llamada del sistema necesario, y ejecutar la interrupción de software.

Tipos de llamadas al sistema

Las llamadas al sistema pueden agruparse a grandes rasgos en cinco categorías principales:

1. Control de procesos
 - carga
 - ejecutar
 - final, abortar
 - crear proceso (por ejemplo, forken los sistemas de tipo Unix, o NtCreateProcessen el Windows NT API nativo)
 - proceso de terminar
 - obtener/establecer atributos de proceso
 - esperar por tiempo, evento de espera, señal de evento
 - asignar, liberar la memoria
2. Gestión de archivos
 - crear el archivo, borrar el archivo
 - abrir, cerrar
 - leer, escribir, reposicionar
 - obtener/establecer atributos de archivo
3. Gestión de dispositivos
 - solicitar dispositivo, liberar dispositivo
 - leer, escribir, reposicionar
 - obtener/establecer atributos del dispositivo
 - conectar o desconectar dispositivos de manera lógica
4. Mantenimiento de la información
 - obtener / establecer la hora o la fecha
 - obtener/establecer la información total del sistema
 - obtener / establecer la metadata de procesos, archivos o dispositivos
5. Comunicación
 - crear, eliminar la conexión de comunicación
 - enviar, recibir mensajes
 - transferencia de la información del estado

- conectar o desconectar dispositivos remotos

Las llamadas al sistema en la mayoría de Unix sistemas se procesan en modo kernel, lo cual se logra cambiando el modo de ejecución del procesador a una más privilegiada, pero ningún proceso de cambio de contexto es necesario - aunque un privilegio se produce cambio de contexto.

El hardware ve el mundo en términos del modo de ejecución de acuerdo con el registro de estado del procesador, y los procesos son una abstracción proporcionada por el sistema operativo. Una llamada al sistema no requiere, en general un cambio de contexto a otro proceso; en su lugar, se procesa en el contexto de cualquiera proceso que lo invocó.

En un proceso multihilo (multithreaded), las llamadas al sistema se pueden hacer de múltiples hilos. El manejo de este tipo de llamadas depende del diseño específico del núcleo del sistema operativo y del entorno en tiempo de ejecución de la aplicación. La siguiente lista muestra modelos típicos seguidos por sistemas operativos:

Modelo Muchos-a-uno:

Todas las llamadas al sistema desde cualquier subproceso de usuario en un proceso son manejados por un solo hilo a nivel de kernel. Este modelo tiene un grave inconveniente, que cualquier llamada de bloqueo al sistema (como esperar la entrada desde el usuario) puede congelar todos los otros hilos. También, puesto que sólo un hilo puede acceder al núcleo a la vez, este modelo puede no utilizar múltiples núcleos de procesador.

Modelo Uno-a-uno:

Cada hebra de usuario se apeg a un hilo distinto a nivel de kernel una llamada al sistema. Este modelo resuelve el problema anterior de bloquear las llamadas al sistema. Se encuentra en todas las principales distribuciones de Linux, macOS, IOS, y versiones recientes de Windows y Solaris.

Modelo Varios a varios:

En este modelo un grupo de subprocesos de usuario se asigna a un grupo de subprocesos del núcleo. Todas las llamadas al sistema de un grupo de subprocesos de usuario son manejadas por los hilos en su correspondiente grupo de hilos del núcleo.

Modelo Híbrido:

Este modelo implementa tanto muchos a muchos y modelo uno a uno dependiendo de la elección hecha por el núcleo. Esto se encuentra en las versiones antiguas de IRIX, HP-UX y Solaris.

Máquinas Virtuales

El concepto de máquina virtual surge con el sistema VM/370 de IBM, en el que se separa las dos funciones que proporciona un sistema de tiempo compartido o multiusuario, como son la multiprogramación y la abstracción del hardware.

Una máquina virtual (VM) es un entorno que funciona como un sistema informático virtual con su propia CPU, memoria, interfaz de red y almacenamiento, pero el cual se crea en un sistema de hardware físico, ya sea on-premise o no. El sistema de software se llama hypervisor, y se encarga de separar los recursos de la máquina del sistema de hardware e implementarlos adecuadamente para que la VM pueda utilizarlos [1].

Las máquinas físicas equipadas con un hypervisor, como la máquina virtual basada en el kernel (KVM), se denominan máquinas host, computadoras host, sistemas operativos host o simplemente hosts. Las diversas máquinas virtuales que usan sus recursos son máquinas guest, computadoras guest, sistemas operativos guest, o simplemente guests. El hypervisor utiliza los recursos informáticos, como la CPU, la memoria y el almacenamiento, como un conjunto de medios que pueden redistribuirse fácilmente entre los guests actuales o en las máquinas virtuales nuevas [1]

Las VM se encuentran aisladas del resto del sistema, pero puede haber varias VM en una sola pieza de hardware, como un servidor. Además, pueden trasladarse entre los servidores host en función de la demanda, o para utilizar los recursos de forma más eficiente. Las VM permiten que se ejecuten varios sistemas operativos diferentes a la vez en una misma computadora, como una distribución de Linux® en una computadora portátil MacOS. Cada sistema operativo funciona de la misma manera en que un SO o una aplicación lo haría normalmente en el hardware del host [1].

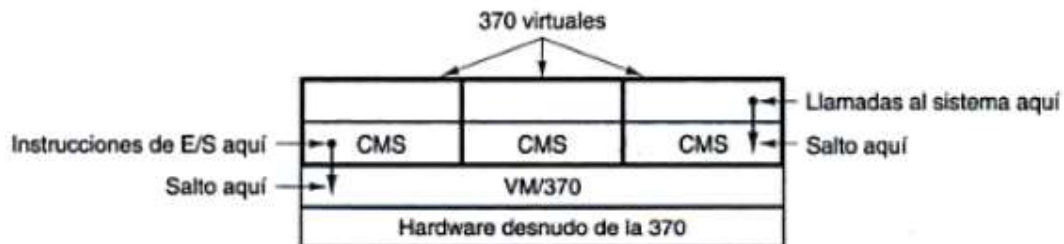


Figura 42: Estructura de una llamada VM/370

Cada máquina virtual puede ejecutar cualquier sistema operativo debido a la circunstancia comentada de que cada una es una copia exacta del hardware. En el caso de la figura anterior se puede apreciar como distintos sistemas operativos monousuarios (CMS, Sistema Monitor de Conversaciones), cuando realizan una llamada al sistema por medio de algún programa, ésta salta al sistema operativo de la propia máquina virtual, no a la VM/370 como hubiera ocurrido en una máquina real, entendiendo que esta sería la capa de software por encima de la capa de hardware de la computadora.

Cómo funcionan las VM

La tecnología de virtualización le permite compartir un sistema con muchos entornos virtuales. El hypervisor gestiona el sistema de hardware y separa los recursos físicos de los entornos virtuales. Los recursos se dividen según las necesidades, desde el entorno físico hasta las VM. Cuando la VM está en ejecución y un usuario o un programa emiten una instrucción que requiere recursos adicionales del entorno físico, el hypervisor programa la solicitud en los recursos del sistema físico para que el sistema operativo y las aplicaciones de la máquina virtual puedan acceder al grupo compartido de recursos físicos [1].

Dentro de los distintos VMM o Hypervisor los más usuales son: VMware en las versiones ESXi 7.0 [15] y Workstation [16], XEN de Citrix [17], Virtual Box de Sun [18], Virtual PC 2007 de Microsoft [19], Hyper V de Microsoft [20], KVM de Redhat.

Estas máquinas virtuales se utilizan hoy en día para diferentes entornos, los más utilizados son los entornos de desarrollo o pruebas, pero también se están utilizando cada vez más para entornos en producción, así se puede encontrar por internet como muchos ISP ofrecen sus servicios de hosting mediante servidores virtuales.

También se utiliza la virtualización para consolidar servidores en una sola máquina con suficientes recursos para ello, así tendremos las máquinas aglutinadas en una misma computadora, a las cuales se podrá hacer una copia de seguridad del sistema completo, ya que este se compone de un solo fichero.

Este último punto junto con la posibilidad que nos ofrecen las máquinas virtuales de ser traspasadas de un host a otro con el mismo tipo de VMM, nos ofrecen un entorno en el que la pérdida de un servicio gestionado por una máquina virtual puede ser mínimo, al poderse restaurar dichas máquinas virtuales de una manera fácil entre los distintos hosts. Otro de los beneficios que proporcionan las máquinas virtuales y por lo que son elegidas es el ahorro de energía. No consume lo mismo una máquina física con 10 máquinas virtuales que 10 máquinas físicas.

Podemos decir por tanto que la virtualización es cada vez más utilizada y más extendida en una amplia variedad de ámbitos, desde un usuario que quiere probar distintos sistemas operativos hasta llegar a algo muy de moda como es el cloud computing (computación en la nube), cuyos servidores están cada vez más virtualizados. Debido a este interés por la virtualización de servidores, se ha decidido utilizar máquinas virtuales para ver no sólo el rendimiento de los sistemas operativos ante un problema dado, sino también como estos responden siendo instalados en máquinas virtuales.



*Hoy puedes brindar servicios de **virtualización de Escritorios, virtualización de aplicaciones**, y un sinnúmero de [productos] **as services**.*

Escritorios virtuales

Los escritorios virtuales son imágenes preconfiguradas de sistemas operativos y aplicaciones en los que el entorno de escritorio está separado del dispositivo físico utilizado para acceder a él. Los usuarios pueden acceder a sus escritorios virtuales de manera remota y con cualquier dispositivo [2].

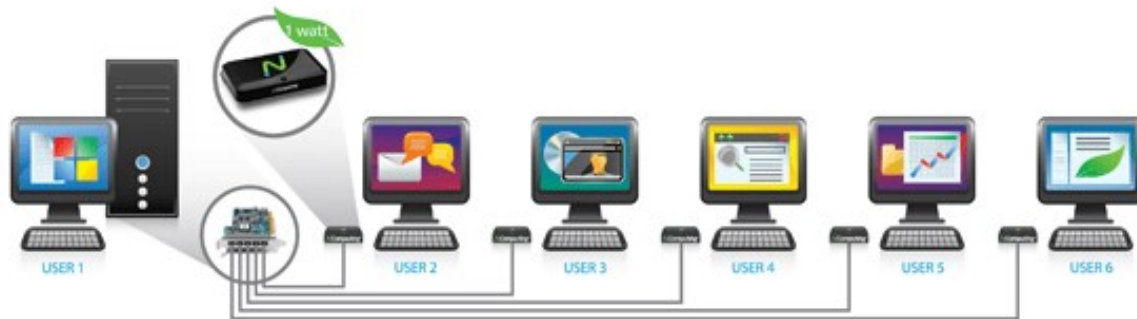


Figura 43: **Virtualización de escritorios**
Fuente: Librestartups

Aplicaciones virtuales

Permita que varias aplicaciones, así como datos de configuración específicos de usuario almacenados en entornos de pruebas, residan de forma segura en el mismo servidor



Figura 44: **Virtualización de aplicaciones**
Fuente: Librestartups



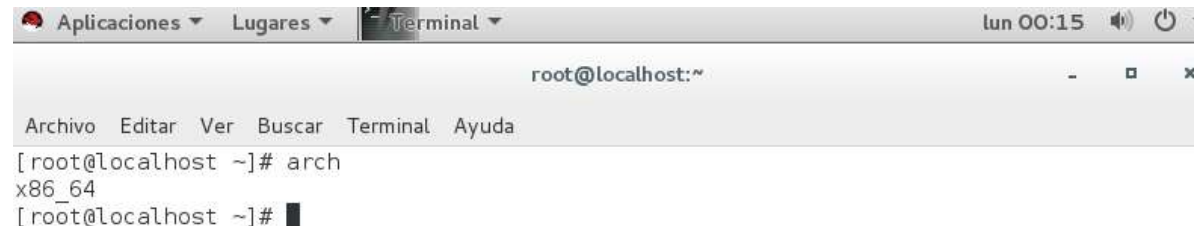
Figura 45: **Virtualización de infraestructura (IaaS)**
Fuente: Librestartups

Comandos básicos Shell Linux

Los comandos son esencialmente los mismos que cualquier sistema UNIX. En las tablas que se presentan a continuación se tiene la lista de comandos más frecuentes.

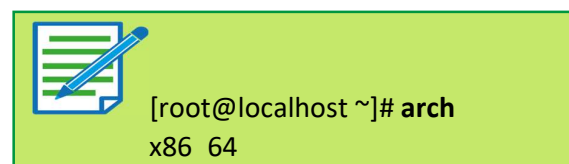
Información del Sistema

A continuación daremos una serie de comandos útiles para conocer aspectos generales del sistema



```
root@localhost:~  
Archivo Editar Ver Buscar Terminal Ayuda  
[root@localhost ~]# arch  
x86_64  
[root@localhost ~]#
```

Figura 46: Resultado de la ejecución del comando **arch**



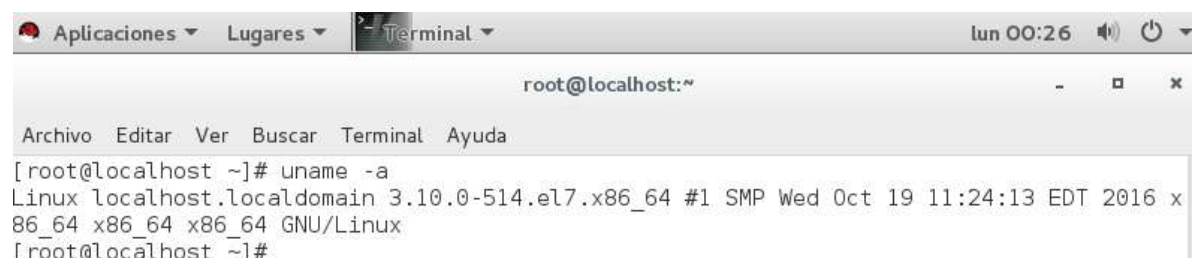
Comando uname [parametros]

La función de este comando es similar al anterior, la única diferencia es que este nos arroja más información del sistema de acuerdo al número de parametros que le pasemos.

Los parametros que podemos usar son:

a. Imprime el nombre kernel, del equipo, versión del kernel, fecha en que fue apagado el sistema por última vez, arquitectura del sistema

- ✓ **s.** Imprime el nombre del kernel
- ✓ **n** .Imprime el nombre del equipo
- ✓ **r** .Imprime versión del kernel
- ✓ **i o p** .Imprime la arquitectura del equipo
- ✓ **o** .Imprime el nombre del sistema operativo



```
root@localhost:~  
Archivo Editar Ver Buscar Terminal Ayuda  
[root@localhost ~]# uname -a  
Linux localhost.localdomain 3.10.0-514.el7.x86_64 #1 SMP Wed Oct 19 11:24:13 EDT 2016 x  
86_64 x86_64 x86_64 GNU/Linux  
[root@localhost ~]#
```

Figura 47: Resultado de la ejecución del comando **uname -a**

```
root@localhost:~# uname -a
Linux localhost 5.8.10-arch1-1.1 #1 SMP PREEMPT Fri, 25 Sep 2020 08:48:44 +0000 i686 GNU/Linux
root@localhost:~# uname -p
unknown
root@localhost:~# uname -o
GNU/Linux
root@localhost:~# uname -r
5.8.10-arch1-1.1
root@localhost:~# uname -s
Linux
root@localhost:~# uname -n
localhost
root@localhost:~# uname -i
unknown
root@localhost:~# _
```

Figura 48: Resultado de la ejecución de los comandos **uname -a, p, o, r, s, n, i**

dmidecode

Lo que hace este comando es leer la información del BIOS directamente y regresar un listado muy completo de todo el hardware encontrado en el equipo. DMI es por Desktop Management interface y lee la información del llamado SMBIOS (System Management BIOS).

dmidecode por defecto ofrece un listado bastante largo y completo, así que si deseas uno más corto o resumido, úsalo con q.

[root@localhost ~]# **dmidecode -q**



```
root@localhost:~# dmidecode -q
BIOS Information
    Vendor: Phoenix Technologies LTD
    Version: 6.00
    Release Date: 04/13/2018
    Address: 0xEA490
    Runtime Size: 88944 bytes
    ROM Size: 64 kB
    Characteristics:
        ISA is supported
        PCI is supported
        PC Card (PCMCIA) is supported
        PNP is supported
        APM is supported
        BIOS is upgradeable
```

Figura 49: Resultado de la ejecución del comando **dmidecode -q**

cat /proc/cpuinfo

Nos muestra la información referente al procesador del sistema

```

root@localhost:~# cat /proc/cpuinfo
processor       : 0
vendor_id     : GenuineIntel
cpu family    : 15
model         : 6
model name    : 0f/06
stepping      : 3
cpu MHz       : 1000.000
cache size    : 6144 KB
physical id   : 0
siblings      : 1
core id       : 0
cpu cores     : 1
apicid        : 0
initial apicid : 0
fdiv_bug      : no
f00f_bug      : no
coma_bug      : no
fpu           : yes
fpu_exception : yes
cpuid level   : 22
wp            : yes
flags         : fpu pse tsc msr cx8 sep pge cmov mmx fxsr sse sse2 constant_tsc tsc_reliable cpuid tsc_known_freq popcnt rdrand
d_hypervisor  :
bugs          : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds swapgs itlb_multihit
bogomips      : 2000.33
clflush size   : 32
cache_alignment : 32
address sizes  : 32 bits physical, 32 bits virtual
power management:

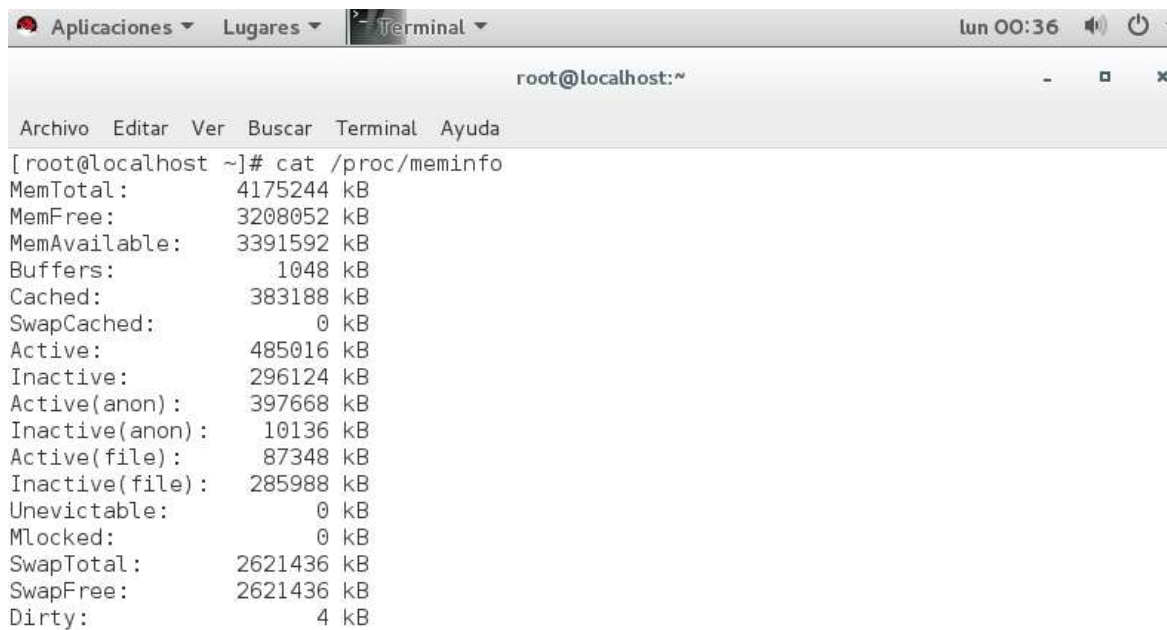
root@localhost:~# _

```

Figura 50: Resultado de la ejecución del comando `cat /proc/cpuinfo`

cat /proc/meminfo

Verifica el uso de la memoria



```

[root@localhost ~]# cat /proc/meminfo
MemTotal:      4175244 kB
MemFree:       3208052 kB
MemAvailable:  3391592 kB
Buffers:        1048 kB
Cached:        383188 kB
SwapCached:      0 kB
Active:        485016 kB
Inactive:      296124 kB
Active(anon):   397668 kB
Inactive(anon): 10136 kB
Active(file):   87348 kB
Inactive(file): 285988 kB
Unevictable:     0 kB
Mlocked:        0 kB
SwapTotal:     2621436 kB
SwapFree:      2621436 kB
Dirty:          4 kB

```

Figura 51: Resultado de la ejecución del comando `cat /proc/meminfo`

cat /proc/swaps

Nos muestra el uso del espacio en memoria SWAP


```

root@localhost:~
Archivo Editar Ver Buscar Terminal Ayuda
[root@localhost ~]# cat /proc/swaps
Filename                                Type              Size      Used      Priority
/dev/dm-1                              partition         2621436   0         -1
[root@localhost ~]#

```

Figura 52: Resultado de la ejecución del comando `cat /proc/swaps`

cat /proc/net/dev

Verifica adaptadores de red y sus estadísticas

```

root@localhost:~
Archivo Editar Ver Buscar Terminal Ayuda
[root@localhost ~]# cat /proc/net/dev
Inter-|   Receive                                   | Transmit
face |bytes  packets errs drop fifo frame compressed multicast|bytes  packets errs
drop fifo colls carrier compressed
ens32: 39959 488 0 0 0 0 0 0 0 0 0 0
      0 0 0 0 0 0 0 0
lo: 43860 516 0 0 0 0 0 0 43860 516 0
   0 0 0 0 0 0 0 0
virbr0-nic: 0 0 0 0 0 0 0 0 0 0 0 0
           0 0 0 0 0 0 0 0
virbr0: 0 0 0 0 0 0 0 0 0 0 0 0
        0 0 0 0 0 0 0 0
[root@localhost ~]#

```

Figura 53: Resultado de la ejecución del comando `cat /proc/net/dev`

cat /proc/mounts

Nos muestra los sistemas de ficheros que se encuentran montados

```

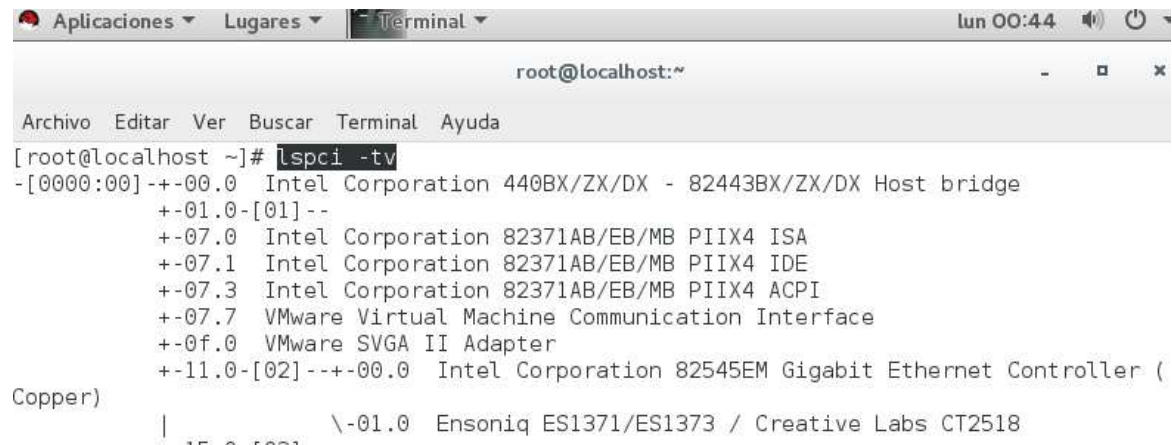
root@localhost:~
Archivo Editar Ver Buscar Terminal Ayuda
[root@localhost ~]# cat /proc/mounts
rootfs / rootfs rw 0 0
sysfs /sys sysfs rw,seclabel,nosuid,nodev,noexec,relatime 0 0
proc /proc proc rw,nosuid,nodev,noexec,relatime 0 0
devtmpfs /dev devtmpfs rw,seclabel,nosuid,size=2071820k,nr_inodes=517955,mode=755 0 0
securityfs /sys/kernel/security securityfs rw,nosuid,nodev,noexec,relatime 0 0
tmpfs /dev/shm tmpfs rw,seclabel,nosuid,nodev 0 0
devpts /dev/pts devpts rw,seclabel,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000 0
0

```

Figura 54: Resultado de la ejecución del comando `cat /proc/mounts`

lspci -tv

Nos lista los dispositivos PCI con lo que dispone el equipo




```
root@localhost:~  
Archivo Editar Ver Buscar Terminal Ayuda  
[root@localhost ~]# lspci -tv  
-[0000:00]--00.0 Intel Corporation 440BX/ZX/DX - 82443BX/ZX/DX Host bridge  
  +-01.0-[01]--  
    +-07.0 Intel Corporation 82371AB/EB/MB PIIX4 ISA  
    +-07.1 Intel Corporation 82371AB/EB/MB PIIX4 IDE  
    +-07.3 Intel Corporation 82371AB/EB/MB PIIX4 ACPI  
    +-07.7 VMware Virtual Machine Communication Interface  
    +-0f.0 VMware SVGA II Adapter  
    +-11.0-[02]---00.0 Intel Corporation 82545EM Gigabit Ethernet Controller (Copper)  
      |  
      +-01.0 Ensoniq ES1371/ES1373 / Creative Labs CT2518
```

Figura 55: Resultado de la ejecución del comando `lspci -tv`

lsusb -tv

Nos lista los dispositivos USB con lo que dispone el equipo

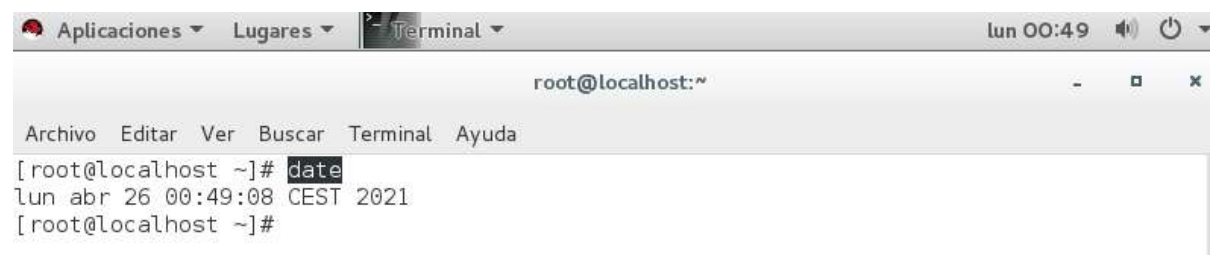


```
root@localhost:~  
Archivo Editar Ver Buscar Terminal Ayuda  
[root@localhost ~]# lsusb -tv  
Jsage: lsusb [options]...  
_list USB devices  
  -v, --verbose  
    Increase verbosity (show descriptors)  
  -s [[bus]:][devnum]  
    Show only devices with specified device and/or  
    bus numbers (in decimal)  
  -d vendor:[product]  
    Show only devices with the specified vendor and  
    product ID numbers (in hexadecimal)
```

Figura 56: Resultado de la ejecución del comando `lsusb -tv`

date

Nos muestra la fecha que tiene registrado el sistema



```
root@localhost:~  
Archivo Editar Ver Buscar Terminal Ayuda  
[root@localhost ~]# date  
lun abr 26 00:49:08 CEST 2021  
[root@localhost ~]#
```

Figura 57: Resultado de la ejecución del comando `date`

En caso de querer modificar la fecha solo se debe de seguir la siguiente sintaxis

date [MesDiaHoraMinutoAño.Segundos]

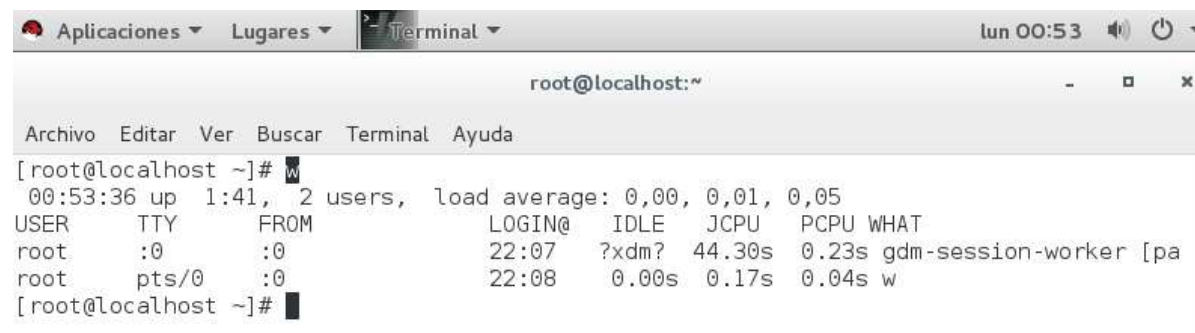
[root@localhost ~]# date 041217002007.00

dmesg

El comando **dmesg** es principalmente usado para mostrar los mensajes que se mostraron en pantalla cuando se arrancó el sistema. Se usa sobre todo para realizar depuraciones al sistema de cómo se están cargando los diversos módulos y componentes al arranque del sistema o ya en ejecución. Debido a lo extenso del sistema, es conveniente re direccionar la salida a un archivo lo cual se puede hacer de la siguiente manera

w

Nos indica los usuarios que se encuentran en el sistema así como lo que hacen en el



```
[root@localhost ~]# w
00:53:36 up 1:41, 2 users, load average: 0,00, 0,01, 0,05
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU WHAT
root      :0                22:07    ?xdm?   44.30s  0.23s  gdm-session-worker [pa
root      pts/0      :0                22:08    0.00s   0.17s  0.04s w
[root@localhost ~]#
```

Figura 58: Resultado de la ejecución del comando w

df -h

Nos reporta el uso de espacio en los discos duros



```
[root@localhost ~]# df -h
S.ficheros          Tamaño Usados  Disp Uso% Montado en
/dev/mapper/rhel-root 22G    4,0G   18G  19% /
devtmpfs             2,0G    0      2,0G  0% /dev
tmpfs                2,0G   476K   2,0G  1% /dev/shm
tmpfs                2,0G    8,9M   2,0G  1% /run
tmpfs                2,0G    0      2,0G  0% /sys/fs/cgroup
/dev/sda1            1014M   173M   842M  17% /boot
tmpfs                408M    8,0K   408M  1% /run/user/0
/dev/sr0              3,6G    3,6G    0 100% /run/media/root/RHEL-7.3 Server.x86_64
[root@localhost ~]#
```

Figura 59: Resultado de la ejecución del comando df -h

ps -xa

Este comando lista los procesos que se están ejecutando en el sistema



```
[root@localhost ~]# ps -xa
PID TTY      STAT   TIME COMMAND
  1 ?        Ss      0:01  /usr/lib/systemd/systemd --switched-root --system --deserial
  2 ?        S        0:00  [kthreadd]
  3 ?        S        0:00  [ksoftirqd/0]
  6 ?        S        0:00  [kworker/u2:0]
  7 ?        S        0:00  [migration/0]
  8 ?        S        0:00  [rcu_bh]
  9 ?        D        0:00  [rcu_sched]
```

Figura 60: Resultado de la ejecución del comando `ps -xa`

- **ls**
Este comando lista los ficheros y directorios (¡Recuerda que en GNU/Linux todo es un fichero!), ejemplo: `ls /home`
Si queremos que nos muestre la información ampliada y en columnas, utilizaremos el parámetro `'-l'`, y si además queremos que nos muestre los elementos ocultos, usaremos `'ls -la /home'`
- **cd**
Con este comando nos podemos mover entre diferentes directorios. Si nos queremos ir a un directorio el particular, ejemplo: `cd /var/cache`
Si en cambio nos encontramos en un directorio, y queremos pasar al inmediatamente superior: `cd..`
O bien ir directamente a otro directorio que esta al mismo nivel que el nuestro: `cd../log`
- **mkdir**
Ya que hablamos de directorios, con este comando podemos crear los que deseemos, como si no hubiera un mañana, ejemplo: `mkdir LaLigaDeLaJusticia`, `mkdir LaLigaDeLaJusticia/Batman`
- **rmdir**
Como si fuéramos un archi enemigo del universo DC, podemos borrar del mapa un directorio, por ejemplo: `rmdir /LaLigaDeLaJusticia/GreenLantern`
- **touch**
Con este comando podemos crear ficheros. Su uso es el siguiente: `touch fichero1 fichero2 fichero3`.
No hace falta que os diga que dichos ficheros están vacíos.
- **rm**
Al igual que con `touch` podemos crear nuevos ficheros, con `rm`, su antítesis (lo siento, a veces me salen estos palabras), podemos eliminarlos de la faz de la tierra.
Es primo hermano del comando comando `'rmdir'`, pero en este caso borra ficheros en vez de directorios, ejemplo: `rm lunes.txt`
Es un comando un poco preguntón, si quieres que se dedique a eliminar sin mala conciencia, sólo debemos añadir el parámetro `'-f'`, y si además quieres que elimine

ficheros y subdirectorios, con ‘-r’, tendrás más que suficiente. Vamos lo que viene a ser un “rm -rf” (Nota: ¡Utilizar este comando con esos parámetros son responsabilidad tuya y solo tuya, eh!)

- *mv*

¿Quieres mover un fichero o un directorio de lugar? ¿Quieres cambiar el nombre de un fichero? ¡Este es tu comando!, ejemplo: mv /facturas/porpagar/factura1.pdf /facturas/yapagadas/
o bien: mv /Avengers/PeterParker.jpg /Avengers/Spider-Man.jpg

- *rename*

Cambia el nombre de un fichero o conjunto de ficheros. Tiene un buen puñado de parámetros interesantes, ejemplo: rename 's/.jpeg/.jpg/' *
De esta manera indicamos que en la ubicación se cambiarán todos los ficheros con extensión “jpeg” por “jpg”

- *man*

Con este comando podemos consultar el manual, de ahí que se llame “man”
Si lo utilizamos seguido del comando a consultar, nos mostrará su entrada en el manual, ejemplo: man touch

- *info*

Es un comando similar al de ‘man’ con información ampliada sobre el comando a consultar, ejemplo: info touch

- *whatis*

Un comando poco conocido, realmente muy útil.
Se encarga de buscar el contenido de la palabra indicada, en una base de datos propia, que contiene breves descripciones de los comandos, ejemplo: whatis man
man (1) - una interfaz de los manuales de referencia electrónicos

- *clear*

Este comando se encarga de borrar la pantalla. No hace mucho más. Para utilizar solo hemos de escribir ‘clear’

- *sudo*

Con este comando nos podemos otorgar los poderes de super usuario, siempre que tengamos permisos para ello, ejemplo: sudo apt-get install nano

- *pwd*

Nos muestra el nombre del directorio de trabajo actual, ejemplo: \$ pwd
/home/david

- *cat*

Muestra el contenido de un fichero dado. Si se utiliza con varios ficheros a la vez, mostrará su contenido de manera secuencial, ejemplo: cat GuiaDelAutoestopistaGalactico-CopiaLegal-eh.txt

- *chmod*
Se encarga de cambiar los permisos de acceso a los ficheros. Le dediqué una entrada a su uso, por si os interesa: “Permisos de archivos en Linux”, ejemplo: `chmod 777 -R /opt/lamp`
- *chown*
Cambia el usuario y grupo propietarios de ficheros, ejemplo: `chown david:familia fotos-de-vacaciones.tar.gz`
- *wget*
Descarga el fichero o página web dada, indicando la URL, ejemplo: `wget https://www.ochobitshacenunbyte.com/`
- *grep*
Busca en uno o más ficheros una cadena determinada de texto. Si encuentra la cadena nos indica donde está. Es un comando muy potente, muy utilizado por DevOps y desarrolladores, ejemplo: `cat /etc/passwd | grep -i davidochobits`
Utilizando el parámetro ‘-i’ ignoramos la diferencia entre mayúsculas y minúsculas.
- *tail*
Imprime las diez últimas líneas de un fichero. Es muy utilizado en la consulta de ficheros de registro, ejemplo: `tail -f -n 20 /var/log/httpd/httpd.log`
De esta manera lista las últimas líneas de registro del fichero `httpd.log`, con el parámetro “-f”, indicamos que queremos ver la actividad del registro “en directo”, con “-n” indicamos que queremos ver siempre las 20 últimas líneas.
- *free*
Muestra la memoria utilizada y disponible en el sistema, incluyendo la swap. Sus parámetros más habituales, por ejemplo, “-m” que nos muestra la información en megabytes, o “-k” que nos la muestra en kilobytes, ejemplo: `free -m`
- *df*
Nos informa de la utilización de disco en un sistema de ficheros. Este comando se encarga de mostrar el espacio usado y del disponible en todos los sistemas de ficheros montados. Mi parámetro favorito es ‘-h’ que muestra información para humanos, vamos más entendible, ejemplo: `df -h`
- *du*
Muestra el espacio estimado ocupado por los ficheros y directorios. Mis parámetros favoritos para este comando son ‘-s’, que nos muestra únicamente el espacio utilizado real, no el de sus subdirectorios, en el caso de un directorio. Y ‘-h’, que muestra el tamaño en megabytes, ejemplo: `du -sh`
- *lsblk*
Muestra información de los dispositivos de bloques, como son los discos duros, volúmenes y grupos de volúmenes, ejemplo: `lsblk -fm`
Con “-f” se muestra información ampliada y con “-m”, muestra información de su propietario y los permisos de lectura.

- *fdisk*
Muestra información de los discos duros y los discos lógicos, llamados particiones, ejemplo: `fdisk -l`
- *ps*
Informa del estado de los procesos. Nos muestra una instantánea de los procesos actuales, ejemplo: `ps -ef`
- *kill*
Se encarga de liquidar un proceso dado, ejemplo: `kill -9 pid-del-proceso`
Su utilizaremos el parámetro “-9”, se encarga de eliminar el proceso y sus procesos hijos. Si queremos ser como Thanos, aunque a lo bruto, podemos utilizar “killall”
- *mount y umount*
Se encarga de montar o desmontar un dispositivo dado sobre el sistema de ficheros, ejemplo: `mount recursnfs:/carpeta /carpeta`
o bien: `umount /carpeta`
- *uname*
Imprime información del sistema, incluyendo la versión del kernel o núcleo y del sistema operativo, ejemplo: `uname -a`
- *uptime*
Indica el tiempo que el sistema encendido, ejemplo: `uptime`
20:04:58 up 0 min, 2 users, load average: 4,26, 1,19, 0,41
- *who*
Nos muestra quien está conectando en el sistema. Importante si lo queremos reiniciar y no queremos que nadie se encuentre con la sorpresa, ejemplo: `who`
root tty1 2018-12-04 20:04
david pts/0 2018-12-04 20:04



Herramientas de entornos digitales

Herramientas SO	Link de descarga
Ubuntu 20.04	http://releases.ubuntu.com/focal/
Redhat	https://access.redhat.com/downloads
Emulador de máquinas virtuales L-W	http://copy.sh/v86/
Software para virtualización VirtualBox	https://www.virtualbox.org/wiki/Downloads

- [1] Redhat, "RedHat," 26 04 2021. [Online]. Available: <https://www.redhat.com/es/topics/virtualization/what-is-a-virtual-machine>.
- [2] vmware, "vmware," 2021. [Online]. Available: <https://www.vmware.com/es/topics/glossary/content/virtual-desktops.html>. [Accessed 26 04 2021].
- [3] W. Stallings, Operating Systems, Internals and Design Princiles, Ninth ed., Malaysia: Pearson, 2018.
- [4] A. Silberschatz, P. B. Galvin and G. Gagne, Operating Systems Concepts, 10th ed., Hoboken: Wiley, 2018.
- [5] Wikipedia, "BootX (Apple)," 15 11 2020. [Online]. Available: [https://en.wikipedia.org/wiki/BootX_\(Apple\)](https://en.wikipedia.org/wiki/BootX_(Apple)).
- [6] Wikipedia, "iBoot," 15 11 2020. [Online]. Available: <https://en.wikipedia.org/wiki/IBoot>.
- [7] Wikipedia, "GNU GRUB," 15 11 2020. [Online]. Available: https://en.wikipedia.org/wiki/GNU_GRUB.
- [8] Wikipedia, "Windows NT 6 startup process," 15 11 2020. [Online]. Available: https://en.wikipedia.org/wiki/Windows_NT_6_startup_process.
- [9] Wikipedia, "NTLDR," 15 11 2020. [Online]. Available: <https://en.wikipedia.org/wiki/NTLDR>.